Microsoft® QuickBASIC

Lernen und Anwenden von Microsoft QuickBASIC

Microsoft®

Microsoft® QuickBASIC
Lernen und Anwenden von Microsoft QuickBASIC

Version 4.5

Für IBM® Personal Computer
Und Kompatible

Microsoft Corporation

Die in diesem Handbuch enthaltenen Angaben sind ohne Gewähr und können ohne weitere Benachrichtigung geändert werden. Die Microsoft Corporation geht hiermit keinerlei Verpflichtungen ein. Die in diesem Handbuch beschriebene Software wird auf Basis eines Lizenzvertrages oder als Einmallizenz (mit der Verpflichtung, sie nicht weiterzugeben) geliefert. Die Software darf nur in Übereinstimmung mit den vertraglichen Bestimmungen verwendet bzw. kopiert werden. Teile dieses Handbuches und/oder der Datenbanken dürfen weder auf elektronische noch mechanische Weise, einschließlich Fotokopien und sonstiger Aufzeichnungen, ohne die schriftliche Genehmigung der Microsoft Corporation vervielfältigt oder übertragen werden.

© Copyright 1988-1989 Microsoft Corporation. Alle Rechte vorbehalten.

Teilen Sie uns bitte Ihre Meinung über die Dokumentation mit und füllen Sie die Dokumentationsbeurteilung am Ende dieses Handbuches aus.

Microsoft®, MS®, MS-DOS®, CodeView®, GW-BASIC® und XENIX® sind eingetragene Warenzeichen der Microsoft Corporation.

Hayes® ist ein eingetragenes Warenzeichen der Hayes Microcomputer Products, Inc.

IBM® und PS/2® sind eingetragene Warenzeichen der International Business Machines Corporation.

Intel® ist ein eingetragenes Warenzeichen der Intel Corporation.

ProKey™ ist ein Warenzeichen der RoseSoft, Inc.

SideKick® und SuperKey® sind eingetragene Warenzeichen der Borland International, Inc.

WordStar® ist ein eingetragenes Warenzeichen der MicroPro International Corporation.

Printed in Ireland: 06

Artikelnr. 06026

Dokumentnr. 8633-450-00-1

Inhaltsverzeichnis — Übersicht

Einleitung

1. Teil: Erste Schritte

1. Kapitel Installieren von Microsoft QuickBASIC
2. Kapitel Verwendung der Menüs und Befehle
3. Kapitel Die QuickBASIC-Fenster
4. Kapitel BASIC für Anfänger

2. Teil: Programmentwicklung in der QuickBASIC-Umgebung

5. Kapitel Das Programm QKARTEN
6. Kapitel Bearbeiten im Arbeitsfenster
7. Kapitel Programmieren mit der On-Line-Hilfe
8. Kapitel Verwendung des Beispielcodes aus der On-Line-Hilfe
9. Kapitel Fehlerbeseitigung während der Programmierung

3. Teil: QuickBASIC, Menüs und Befehle

10. Kapitel Anwendung von QuickBASIC
11. Kapitel Das Menü Datei
12. Kapitel Verwendung des Editors
13. Kapitel Das Menü Bearbeiten
14. Kapitel Das Menü Ansicht
15. Kapitel Das Menü Suchen
16. Kapitel Das Menü Ausführen
17. Kapitel Grundkonzepte und Techniken der Fehlerbeseitigung
18. Kapitel Das Menü Debug
19. Kapitel Das Menü Aufrufe (nur im vollständigen Menü)
20. Kapitel Das Menü Optionen
21. Kapitel Das Menü Hilfe

Glossar

Anhang Referenzzeilen: Tastenkurzkombinationen für die Befehlsflächen

Index

Inhaltsverzeichnis

Einleitung

Sofortige Kontrolle xv

On-Line-Hilfe xvi

Hardwareanforderungen xvi

QuickBASIC-Dokumentation xvi
Gedruckte Dokumentation xvii
Angeschlossene Dokumentation xviii

Typographische Konventionen xix

Dokumentationsbeurteilung xxii

1. Teil: Erste Schritte

1 Installieren von Microsoft QuickBASIC

Das Programm SETUP 1.1

Das Setup-Hauptmenü 1.2

Die Installation 1.2

QB-Express 1.3
Programmieren mit QuickBASIC 1.3
Für den Programmieranfänger 1.3
Für den fortgeschrittenen BASIC-Programmierer 1.4

Die Arbeit mit der Maus 1.4

2 Verwendung der Menüs und Befehle

Starten von QuickBASIC 2.1

Die QuickBASIC-Umgebung 2.2
- Die Menüzeile 2.2
- Die Referenzzeile 2.3
- Das Öffnen der Menüs 2.4

Die On-Line-Hilfe von QuickBASIC 2.5
- Verwendung der On-Line-Hilfe 2.5
- Das Hilfsmenü 2.5

Die Befehlsauswahl 2.7

Verwendung der Dialogfelder 2.9
- Zusammensetzung eines Dialogfeldes 2.10
- Anzeige eines Dialogfeldes 2.11
- Andere Einrichtungen eines Dialogfeldes 2.12

Die direkte Befehlseingabe 2.13
- Tasten zur direkten Befehlseingabe 2.14
- Weitere Tastenkombinationen 2.14

Verlassen von QuickBASIC 2.15

Weitere Informationen 2.16

3 Die QuickBASIC-Fenster

In den Kurzmenüs verfügbare Fenster 3.1

Das Arbeitsfenster 3.3
- Das Laden von Programmen in das Arbeitsfenster 3.3
- Bewegen des Cursors 3.5
- Rollen des Textes 3.6
- Starten eines neuen Programms 3.6
- Ändern der Fenstergröße 3.8

Andere QuickBASIC-Fenster 3.8
- Bewegen des Cursors zwischen den Fenstern 3.8
- Codeausführung im Direktfenster 3.9
- Überwachen der Variablen mit Hilfe des Debug-Fensters 3.11
- Das Hilfsfenster 3.12
  - Kontextabhängige Hilfe 3.12
  - Hyperverbindungen 3.13

Verlassen der QuickBASIC-Umgebung und Speichern der Programme 3.15

Weitere Informationen 3.16

4 BASIC für Anfänger

Programmdefinition 4.2

Kommentare 4.2

Darstellung von Wörtern und Zahlen auf dem Bildschirm 4.2

Variablen 4.3
- Speichern von Daten 4.4
- Variablentypen 4.5
  - Ganzzahlige Variablen 4.5
  - Gleitkommavariablen 4.5
  - Zeichenkettenvariablen 4.6

Zuweisung von Werten an Variablen 4.7

Berechnungen 4.8
- Ganzzahlige Division und Restoperator 4.9
- Vorrang der Operationen 4.9
- Mathematische Funktionen 4.10

Ausdrücke 4.12

Ausgabe von Variablen und Ausdrücken 4.12

Dateneingabe anhand der INPUT-Anweisung 4.15

Datenfelder 4.16
- Deklarieren von Datenfelder 4.17
- Angabe der Datenfeldelemente 4.17

Beim Fällen von Entscheidungen verwendete logische Vergleiche 4.17
- Vergleichsoperatoren 4.18
- Boolesche Ausdrücke 4.19
- Zusammengesetzte Ausdrücke 4.20

Verwendung logischer Anweisungen zur Ablaufsteuerung des Programms 4.21

Wiederholung von Programmoperationen 4.23
- Die FOR...NEXT-Schleife 4.23
- Die DO...LOOP-Schleife 4.25
  - Die DO WHILE LOOP-Schleife 4.26
  - Die DO UNTIL-Schleife 4.27

Schreiben eines einfachen BASIC-Programms 4.29

Weitere Informationen 4.30
- BASIC-Bücher 4.30
- Bücher über DOS 4.30

2. Teil: Programmentwicklung in der QuickBASIC-Umgebung

5 Das Programm QKARTEN

Der Aufbau des Programms QKARTEN 5.1

Laden eines Programms beim Starten von QuickBASIC 5.2

QKARTEN — Ein kurzer Überblick 5.2
- Das Programm QKARTEN 5.3
- Vereinbarungen und Definitionen 5.4
- Kommentare 5.4
- Nach der Anweisung END stehende Anweisungen 5.5
- Aufrufen der QKARTEN-Prozeduren aus dem Direktfenster 5.5
- Unterbrechen einer bedingungslosen Schleife in QKARTEN 5.6
- Der Modul-Ebenen-Code 5.7
- Strukturierte Programmierung mit Prozeduren 5.7
- Die Struktur des Programms 5.9
- Definieren einer Prozedur in QKARTEN 5.11

Speichern des bearbeiteten Textes 5.12

Weitere Informationen 5.14

6 Bearbeiten im Arbeitsfenster

Der intelligente Editor 6.1
- Das automatische Formatieren 6.2
- Syntaxüberprüfung 6.4
- Während des Programmablaufs entdeckte Fehler 6.5
  - Hilfe für Fehlermeldungen 6.6
  - Überschreiben des Fehlers 6.7
- Automatische Prozedur-Vereinbarungen 6.8

Die Arbeit mit ausgewählten Textblöcken 6.9
- Ausschneiden und Kopieren des ausgewählten Texts 6.9
- Einfügen eines Textblocks 6.10
- Manipulieren des ausgewählten Texts 6.11

Suchen und Ersetzen von Text 6.12
- Definieren einer symbolischen Konstante 6.13
- Ersetzen mehrerer Textvorkommen 6.14
- Überprüfen der Arbeit 6.16

Weitere Informationen 6.17

7 Programmieren mit der On-Line-Hilfe

Aufbau von Anweisungen anhand der On-Line-Hilfe 7.1
On-line-Hilfe für Schlüsselwörter 7.2
Hyperverbindungen in der On-Line-Hilfe 7.3
On-Line-Hilfe für Programmsymbole 7.5

Drucken von Anzeigen aus der On-Line-Hilfe 7.8

Weitere Informationen 7.10

8 Verwendung des Beispielcodes aus der On-Line-Hilfe

Kopieren von Beispielcode aus der On-Line-Hilfe 8.1

Einrückung eines Codeblocks 8.4

Kopieren umfangreicher Code-Blöcke aus der On-Line-Hilfe 8.4

Bearbeiten des aus der On-Line-Hilfe kopierten Codeblocks 8.8

Abschluß des QKARTEN-Codes 8.11

Verwendung des Programms QKARTEN 8.12

Weitere Informationen 8.13

9 Fehlerbeseitigung während der Programmierung

Befehle zum Debuggen 9.1
Die Befehle des Menüs Debug 9.2
Die Befehle zum Debuggen im Menü Ausführen 9.2
Bei der Fehlerbeseitigung verwendete Funktionstasten 9.3

Debuggen einer Prozedur 9.3
Kennenlernen der Prozeduren 9.6
Fortsetzung eines unterbrochenen Programms 9.6

Isolieren eines Fehlers 9.8

Schließen des Debug-Fensters 9.12

Automatische Prozedurvereinbarungen 9.12

Erstellung einer selbständig ausführbaren Datei 9.13

Kennenlernen der weiteren QuickBASIC-Menüpunkte 9.14

Weitere Informationen 9.15

3. Teil: QuickBASIC, Menüs und Befehle

10 Anwendung von QuickBASIC

10.1 Der Aufruf von QuickBASIC 10.1
- 10.1.1 Der QB-Befehl 10.2
- 10.1.2 Die Option /NOHI 10.4
- 10.1.3 Die QuickBASIC-Anzeige 10.4

10.2 Öffnen der Menüs und Auswählen der Befehle 10.8
- 10.2.1 Anwendung der Tastatur 10.8
- 10.2.2 Einsatz der Tastenkombinationen zur direkten Befehlseingabe 10.10

10.3 Verwendung der Dialogfelder 10.12

10.4 Verwendung der Fenster 10.15
- 10.4.1 Die verschiedenen Fenster 10.16
- 10.4.2 Teilen des Arbeitsfensters (nur vollständiges Menü) 10.16
- 10.4.3 Wechseln des aktiven Fensters 10.17
- 10.4.4 Verändern der Fenstergröße 10.18
- 10.4.5 Rollen im aktiven Fenster 10.18

10.5 Verwendung des Direktfensters 10.20
- 10.5.1 Unzulässige Anweisungen 10.22
- 10.5.2 Durchführung von Berechnungen 10.22
- 10.5.3 Testen der Bildschirmausgabe 10.22
- 10.5.4 Prozeduraufruf 10.23
- 10.5.5 Ändern der Variablenwerte 10.24
- 10.5.6 Simulieren von Laufzeitfehlern 10.25

10.6 Verwendung des Debug-Fensters 10.26

10.7 Verwendung der Maus 10.27

10.8 Verwendung der On-Line-Hilfe 10.29
- 10.8.1 Hilfseinrichtungen 10.29
- 10.8.2 Die Hyperverbindungen 10.30
- 10.8.3 Bewegen innerhalb des Hilfsfensters 10.32
- 10.8.4 Die Hilfsdateien 10.32
- 10.8.5 Festplatten-System 10.33
- 10.8.6 System mit Diskettenlaufwerken 10.33

11 Das Menü Datei

11.1 Der Befehl Neues Programm 11.2

11.2 Der Befehl Programm laden 11.4
- 11.2.1 Angabe einer Datei 11.5
- 11.2.2 Auflisten des Verzeichnisinhalts 11.6

11.3 Der Befehl Zusammenführen (nur im vollständigen Menü) 11.7
11.4 Der Befehl Speichern (nur im vollständigen Menü) 11.9
11.5 Der Befehl Speichern unter 11.11
11.6 Der Befehl Alles Speichern (nur im vollständigen Menü) 11.12
11.7 Der Befehl Datei erstellen (nur im vollständigen Menü) 11.13
11.8 Der Befehl Datei laden (nur im vollständigen Menü) 11.15
11.9 Der Befehl Datei entfernen (nur im vollständigen Menü) 11.17
11.10 Der Befehl Drucken 11.19
11.11 Der Befehl Betriebssystem (nur im vollständigen Menü) 11.20
11.12 Der Befehl Ende 11.21

12 Verwendung des Editors

12.1 Die Texteingabe 12.1
12.2 Die Textauswahl 12.2
12.3 Die Texteinrückung 12.2
12.4 Verwendung von Punkten zur Textmarkierung 12.3
12.5 Der intelligente Editor 12.4
12.5.1 Einschalten des intelligenten Editors 12.4
12.5.2 Die automatische Syntaxüberprüfung 12.4
12.5.3 Die Fehlermeldungen 12.5
12.5.4 Die automatische Formatierung 12.8
12.6 Eingabe von Sonderzeichen 12.8
12.7 Zusammenfassung der Editierbefehle 12.9

13 Das Menü Bearbeiten

13.1 Der Zwischenspeicher 13.2
13.2 Der Befehl Rückgängig (nur im vollständigen Menü) 13.2
13.3 Der Befehl Ausschneiden 13.3
13.4 Der Befehl Kopieren 13.4
13.5 Der Befehl Einfügen 13.5
13.6 Der Befehl Löschen (nur im vollständigen Menü) 13.6
13.7 Der Befehl Neue SUB (nur im vollständigen Menü) 13.7
13.7.1 Erstellen einer neuen SUB-Prozedur 13.8
13.7.2 Standard-Datentypen für Prozeduren 13.9
13.7.3 Ändern des Standardtyps einer Prozedur 13.10
13.7.4 Speichern und Benennen von Prozeduren 13.10
13.8 Der Befehl Neue FUNCTION (nur im vollständigen Menü) 13.11

14 Das Menü Ansicht

14.1 Der Befehl SUBs 14.2

14.2 Der Befehl Nächste SUB (nur im vollständigen Menü) 14.5

14.3 Der Befehl Teilen (nur im vollständigen Menü) 14.6

14.4 Der Befehl Nächste Anweisung (nur im vollständigen Menü) 14.7

14.5 Der Befehl Ausgabebildschirm 14.8

14.6 Der Befehl Bearbeiten Include-Datei (nur im vollständigen Menü) 14.9
- 14.6.1 Verschachteln von Include-Dateien 14.10
- 14.6.2 Auffinden von Include-Dateien 14.10

14.7 Der Befehl Anzeigen Include-Datei 14.11

15 Das Menü Suchen

15.1 Definieren des Zieltexts 15.1

15.2 Der Befehl Suchen 15.2

15.3 Der Befehl Markierter Text (nur im vollständigen Menü) 15.4

15.4 Der Befehl Weitersuchen (nur im vollständigen Menü) 15.5

15.5 Der Befehl Ändern 15.6

15.6 Der Befehl Marke (nur im vollständigen Menü) 15.8

16 Das Menü Ausführen

16.1 Der Befehl Start 16.2

16.2 Der Befehl Neustart 16.3

16.3 Der Befehl Weiter 16.4

16.4 Der Befehl Ändere COMMAND$ (nur im vollständigen Menü) 16.5

16.5 Der Befehl EXE-Datei erstellen 16.6
- 16.5.1 Die Erstellung ausführbarer Dateien 16.6
- 16.5.2 Quick-Bibliotheken und ausführbare Dateien 16.8
- 16.5.3 Ausführbare Dateitypen 16.9
  - 16.5.3.1 Programme mit Zugriff auf das Laufzeitmodul
  - 16.5.3.2 Selbständige Programme
- 16.5.4 Fehlerprüfung während der Laufzeit ausführbarer Dateien 16.10
- 16.5.5 Gleitkommaarithmetik in ausführbaren Dateien 16.11

16.6 Der Befehl Bibliothek erstellen (nur im vollständigen Menü) 16.12
- 16.6.1 Entfernen und Laden von Modulen 16.12
- 16.6.2 Die Erstellung von Bibliotheken 16.13

16.7 Der Befehl Hauptmodul bestimmen (nur im vollständigen Menü) 16.14
- 16.7.1 Ändern des Hauptmoduls 16.15
- 16.7.2 Die MAK-Datei 16.15

17 Grundkonzepte und Techniken der Fehlerbeseitigung

17.1 Debuggen mit QuickBASIC 17.1

17.2 Fehlervermeidung anhand von QuickBASIC 17.2

17.3 Die QuickBASIC-Einrichtungen zum Debuggen 17.3
- 17.3.1 Die Ablaufverfolgung (nur im vollständigen Menü) 17.4
- 17.3.2 Haltepunkte und Stoppbedingungen 17.4
- 17.3.3 Das Anzeigen von Variablen und Ausdrücken 17.5
- 17.3.4 Das Debug-Fenster 17.5
- 17.3.5 Das Direktfenster 17.6
- 17.3.6 Andere Debug-Einrichtungen 17.6

18 Das Menü Debug

18.1 Der Befehl Variable anzeigen 18.2

18.2 Der Befehl Aktuellen Wert anzeigen 18.5

18.3 Der Befehl Stoppbedingung (nur im vollständigen Menü) 18.7

18.4 Der Befehl Anzeigevariable löschen 18.8

18.5 Der Befehl Alle Anzeigevariablen löschen (nur im vollständigen Menü) 18.10

18.6 Der Befehl Verfolgen ein (nur im vollständigen Menü) 18.11

18.7 Der Befehl Rückverfolgen ein (nur im vollständigen Menü) 18.12
- 18.7.1 Verfolgen rückwärts 18.12
- 18.7.2 Verfolgen vorwärts 18.13

18.8 Der Befehl Haltepunkt ein/aus 18.13

18.9 Der Befehl Alle Haltepunkte löschen 18.14

18.10 Der Befehl Halt bei Fehler (nur im vollständigen Menü) 18.15

18.11 Der Befehl Nächste Anweisung festlegen 18.16

19 Das Menü Aufrufe (nur im vollständigen Menü)

19.1 Die Anwendung des Menüs Aufrufe 19.2

19.2 Aktive Prozeduren 19.4

20 Das Menü Optionen

20.1 Der Befehl Bildschirm 20.2

20.2 Der Befehl Suchpfade festlegen 20.3

20.3 Der Befehl Rechte Maustaste (nur im vollständigen Menü) 20.5

20.4 Der Befehl Syntaxüberprüfung (nur im vollständigen Menü) 20.6

20.5 Der Befehl Vollständiges Menü 20.7

21 Das Menü Hilfe

21.1 Der Befehl Index 21.1

21.2 Der Befehl Themen 21.2

21.3 Der Befehl Begriff 21.3

21.4 Der Befehl Überblick 21.5

Glossar

Anhang Referenzzeilen: Tastenkurzkombinationen für die Befehlsflächen

Index

Einleitung

Microsoft® QuickBASIC stellt eine Programmierumgebung dar, die alle für das Schreiben, Bearbeiten, Ausführen und Debuggen eines Programms erforderlichen Werkzeuge enthält. Diese Programmierwerkzeuge sind in einer leistungsfähigen Version der BASIC-Programmiersprache eingegliedert und verfügen über eine On-Line-Hilfe, die sowohl Umgebung als auch Programmiersprache ausführlich erläutert.

Sofortige Kontrolle

Microsoft QuickBASIC beschleunigt den Programmier- und Lernprozess, indem es dem Benutzer eine sofortige Kontrolle der eigenen Ideen bietet. Beim Schreiben eines Programms wird "Code" eingegeben (eine Folge von QuickBASIC-Anweisungen), der die gewünschte Programmaktion beschreibt. QuickBASIC überprüft die Gültigkeit jeder Zeile bereits während der Eingabe und übersetzt den Code sofort in eine vom Computer ausführbare Form. Falls der Code Fehler enthält, die seine Übersetzung verhindern, gibt QuickBASIC den Fehler an und hilft bei dessen Korrektur. Anschließend kann der korrekte Code durch eine einfache Tastenbetätigung sofort gestartet werden. Wenn der Code die erwartete Aktion nicht durchführt, können Fehler der Codelogik anhand der komplexen QuickBASIC-Einrichtungen zur Fehlerbeseitigung verfolgt und korrigiert werden. Dem Benutzer kommt die Geschwindigkeit und Leistungsfähigkeit einer kompilierten Programmiersprache zugute, ohne die langwierigen Vorgänge der getrennten Bearbeitung, Kompilierung, Ausführung und Fehlerbeseitigung.

On-Line-Hilfe

Die On-Line-Hilfe von Microsoft QuickBASIC besteht aus zwei Teilen. Der Microsoft QB-Ratgeber enthält Erklärungen und ausführbare Beispiele für jede Anweisung der QuickBASIC-Programmiersprache. Es steht dem Benutzer auch eine direkte Hilfe zu allen Fragen zur QuickBASIC-Umgebung und für sämtliche im Programm definierte Symbole sowie Fehlermeldungen zur Verfügung.

Dieses Handbuch zeigt, wie Informationen zu folgenden Themen von der On-Line-Hilfe angefordert werden:

- QuickBASIC-Anweisungen und -Funktionen.
- Menüs, Dialogfeldeingaben und Optionen.
- Für die Bearbeitung und Fehlerbeseitigung bestimmte Sondertasten und Tastenkombinationen.
- Fehlerkorrektur im Programm.
- Im Programm definierte Symbole.

Hardwareanforderungen

Microsoft QuickBASIC erfordert einen IBM Personal Computer oder einen IBM-PC kompatiblen Computer mit einem Arbeitsspeicher von mindestens 384 Kilobyte (K) und einer Disketten-Laufwerkkapazität von mindestens 720 K. Für eine optimale Leistung werden eine Festplatte und ein Arbeitsspeicher von 640K empfohlen.

Die QuickBASIC-Umgebung unterstützt jede zu der Microsoft Mouse kompatible Maus. Nähere Informationen zur Verwendung der Maus mit QuickBASIC sind Abschnitt 10.7 zu entnehmen.

Hinweis In diesem Handbuch bezeichnet der Begriff "DOS" sowohl MS-DOS® als auch das Betriebssystem IBM Personal Computer DOS. Der Name eines bestimmten Betriebssystems wird bei dem Hinweis auf spezifische Eigenschaften des betreffenden Systems verwendet.

QuickBASIC-Dokumentation

Das Microsoft QuickBASIC-Paket enthält zwei Handbücher, eine ausführliche On-Line-Hilfe, Beispielprogramme und weiteres angeschlossenes Dokumentationsmaterial. Die nächsten Abschnitte beschreiben den Inhalt dieses Pakets.

Gedruckte Dokumentation

Das Microsoft QuickBASIC-Paket enthält zwei Handbücher. Dieses Handbuch behandelt die QuickBASIC-Programmierumgebung. Die ersten beiden Teile bilden das Lehrmaterial zur Verwendung von QuickBASIC. Der letzte Teil bezieht sich auf QuickBASIC-Menüs und -Befehle. In der folgenden Liste werden diese Teile eingehender aufgeführt:

Teil	***Inhalt***
"Erste Schritte"	Allgemeine Beschreibung der Anwendung von QuickBASIC und eine kurze Einführung in die BASIC-Programmierung.
"Programmentwicklung in der QuickBASIC-Umgebung"	Einführung in die Kurzmenü-Version der QuickBASIC-Umgebung. Dieses Lehrmaterial gibt dem Benutzer die Möglichkeit, Bearbeitung, Fehlerbeseitigung und On-Line-Hilfe zu üben und weist ihn in der Erstellung der ersten QuickBASIC-Anwendung, *qkarten.bas*, an.
"QuickBASIC, Menüs und Befehle"	Referenz der QuickBASIC-Umgebung, die ausführliche Informationen über jeden Befehl der vollständigen Menüs von QuickBASIC erteilt. Diese Menüs gewährleisten die vollständige mehrmodulige Funktionalität von QuickBASIC. Erweitern Sie Ihre Kenntnisse mit Hilfe dieses Abschnitts, nachdem Ihnen die QuickBASIC-Grundbegriffe vertraut sind.
"Glossar"	Definitionen der in der Dokumentation verwendeten Begriffe.

Dieses Handbuch umfaßt folgende, zu Beginn der Programmierung mit QuickBASIC notwendige Informationen:

- Die QuickBASIC-Umgebung.
- Die QuickBASIC-Implementierung der BASIC-Sprache.
- Die Programmiergrundsätze.

Wenn Ihnen das Programmieren, die BASIC-Sprache oder beides unbekannt ist, empfiehlt es sich, sowohl die Umgebung als auch die QuickBASIC-Version der BASIC-Sprache und allgemeine Programmiertechniken zu erlernen. Sie sollten in diesem Fall mit den Kapiteln 1 bis 4 des 1. Teils, "Erste Schritte", beginnen. Wenn Ihnen das Programmieren in BASIC bereits geläufig ist, werden Sie in der Lage sein, nach Durchlesen der Kapitel 1 bis 3 des 1. Teils mit dem Programmieren zu beginnen.

Im 2. Teil, "Die Programmentwicklung in der QuickBASIC-Umgebung", werden Sie einige der interessantesten Eigenschaften der QuickBASIC-Umgebung, wie zum Beispiel den komplexen Editor, die On-Line-Hilfe und die Fehlerbeseitigung, anwenden. Im Laufe dieses Abschnitts werden Sie ebenfalls die Gelegenheit haben, ein nützliches Datenbank-Anwendungsprogramm, *qkarten.bas*, aufzubauen.

Das QuickBASIC-Paket enthält ein weiteres Handbuch, *Programmieren in BASIC*, mit folgender Thematik:

- Spezifische, praktische Programmierthemen, die jeweils durch ausführliche Programmierbeispiele veranschaulicht werden.
- Die Funktion und Syntax sämtlicher QuickBASIC-Anweisungen und Funktionen, nach Aufgaben tabellenförmig gruppiert.
- Informationen über verschiedene Themen, wie die Konversion interpretierter BASIC-Programme in QuickBASIC, Änderungen der Version 4.5 im Vergleich zu früheren QuickBASIC-Versionen und Kompilieren und Binden aus DOS.

Mit *Programmieren in BASIC* verfügen Sie über gründliche Informationen zu Programmierthemen und eine Übersicht der BASIC-Sprache.

Angeschlossene Dokumentation

Wie aus der nachstehenden Liste ersichtlich, bietet Microsoft QuickBASIC eine verständliche Hilfe und mehrere Diskettendateien, die zusätzliche Informationen und Programmierbeispiele beinhalten:

Dokumentation	***Ziel***
On-Line-Hilfe	Enthält sämtliche für das Programmieren in QuickBASIC erforderliche Informationen. Werden Hilfsbefehle in der Umgebung benutzt, sucht QuickBASIC nach der gewünschten Information in mehreren Dateien. Die Datei *qb45qck.hlp* enthält Zusammenfassungen und Syntaxbeschreibungen aller QuickBASIC-Anweisungen und -Funktionen. Die Datei *qb45advr.hlp* enthält umfassende Informationen über die Verwendung jeder Anweisung und Funktion sowie einen kopier- und anwendbaren Beispielcode. Die Datei *qb45ener.hlp* erklärt alle Menüs, Befehle, Dialogfelder und Fehlermeldungen.
liesmich.doc	Eine Datei, die die nach dem Druck des Handbuches an QuickBASIC vorgenommenen Änderungen vorstellt.

Dokumentation	*Ziel*
paket.lst	Eine Datei, die alle auf den QuickBASIC-Originaldisketten vorhandenen Dateien beschreibt.
lernen.com	Microsoft QB-Express, ein Lernprogramm, das in die QuickBASIC-Umgebung einführt.
Beispielprogramme	Alle in *Programmieren in BASIC* aufgeführten Programme sowie weitere Beispielprogramme. Die Datei *packing.lst* enthält Informationen, in welchen Verzeichnissen sie zu finden sind. Zahlreiche dieser Programme demonstrieren fortgeschrittene Programmiertechniken- und algorithmen.

Typographische Konventionen

In diesem Handbuch werden folgende typographische Konventionen verwendet:

Darstellung	*Beschreibung*
COPY, LINK, /X	Großbuchstaben zeigen Befehle auf DOS-Ebene an. Großbuchstaben werden auch für Befehlszeilen-Optionen benutzt, es sei denn, die Anwendung nimmt nur Kleinbuchstaben an.
SUB, IF, LOOP, PRINT, WHILE, TIME$	Fettgedruckte Großbuchstaben zeigen sprachspezifische Schlüsselwörter mit besonderer Bedeutung in Microsoft BASIC an. Diese Schlüsselwörter sind ein notwendiger Bestandteil der Anweisungs-Syntax, es sei denn, sie sind, wie nachfolgend beschrieben, in Klammern eingeschlossen. Beim Schreiben von Programmen muß die Schlüsselworteingabe genau wie gezeigt erfolgen. Es können jedoch dabei sowohl Groß- als auch Kleinbuchstaben verwendet werden.

Darstellung	*Beschreibung*
qb.lib, add.exe	Kursive Kleinbuchstaben geben Dateinamen oder -erweiterungen an.
`CALLNewProc (Arg1!, Var2%)`	Diese Schriftart wird zur Darstellung von Programmbeispielen, Programmausgabe und Fehlermeldungen innerhalb des Textes benutzt.

`'$INCLUDE:'BC.BI'`
`.`
`.`
`.`
`CHAIN "PROG1"`
`END`
	Eine dreipunktige Spalte zeigt an, daß ein Programmteil absichtlich ausgelassen wurde.

`' Führe eine Übergabe` `' durch`	Der Apostroph (rechtes einzelnes Anführungszeichen) kennzeichnet den Anfang eines Kommentars in Beispielprogrammen.
Dateiangabe	Begriffe in Kursivschrift sind Platzhalter für einzugebende Informationen, wie Dateinamen. Die Kursivschrift wird gelegentlich auch zum Unterstreichen einer Textpassage benutzt.
[*Wahlweiser Begriff*]	Zwischen eckigen Klammern stehende Begriffe sind optional.
{*Wahl1* | *Wahl2*}	Geschweifte Klammern und vertikale Balken zeigen eine Auswahl zwischen zwei oder mehreren Begriffen an. Der Benutzer muß eine Wahl treffen, es sei denn, daß sämtliche Begriffe auch in eckigen Klammern stehen.
Wiederholende Elemente...	Drei nach einem Begriff stehende Punkte deuten darauf hin, daß mehrere Begriffe derselben Form auftreten können.
Unterstrich (_)	Der Unterstrich kennzeichnet in Beispielprogrammen, daß die BASICzeile hier nicht endet, sondern in der nächsten Zeile im Handbuch fortgesetzt wird.

Darstellung	*Beschreibung*
ALT+F1	Kleingedruckte Großbuchstaben werden für die Bezeichnungen von Tasten und Tastenkombinationen wie EINGABETASTE und STRG+R benutzt.
	Ein Pluszeichen (+) weist auf eine Tastenkombination hin. Zum Beispiel weist STRG+E den Benutzer an, die E-TASTE bei niedergedrückter STRG-TASTE zu betätigen.
	Die manchmal mit einem gekrümmten Pfeil gekennzeichnete Wagenrücklauftaste wird als EINGABETASTE bezeichnet.
	Die im rechten Tastenfeld vorhandenen Cursor-Bewegungstasten ("Pfeiltasten") werden RICHTUNGSTASTEN genannt. Einzelne RICHTUNGSTASTEN werden entweder durch die Richtung des Pfeiles auf der Taste bezeichnet (NACH LINKS, NACH RECHTS, NACH OBEN, NACH UNTEN) oder durch die Bezeichnung auf der Taste (BILD↑, BILD↓).
	Die in diesem Handbuch verwendeten Tastenbezeichnungen entsprechen den auf den Tasten des IBM-Personal Computers stehenden Bezeichnungen. Andere Computer können abweichende Bezeichnungen benutzen.
"Definierter Begriff"	Anführungsstriche kennzeichnen normalerweise einen neuen im Text definierten Begriff.
Video Graphics Array (VGA)	Abkürzungen werden üblicherweise ausgeschrieben, wenn sie zum ersten Mal im Text auftreten.

Die untenstehende Syntax (der Anweisungen "**LOCK...UNLOCK**") veranschaulicht viele der typographischen Konventionen dieses Handbuches:

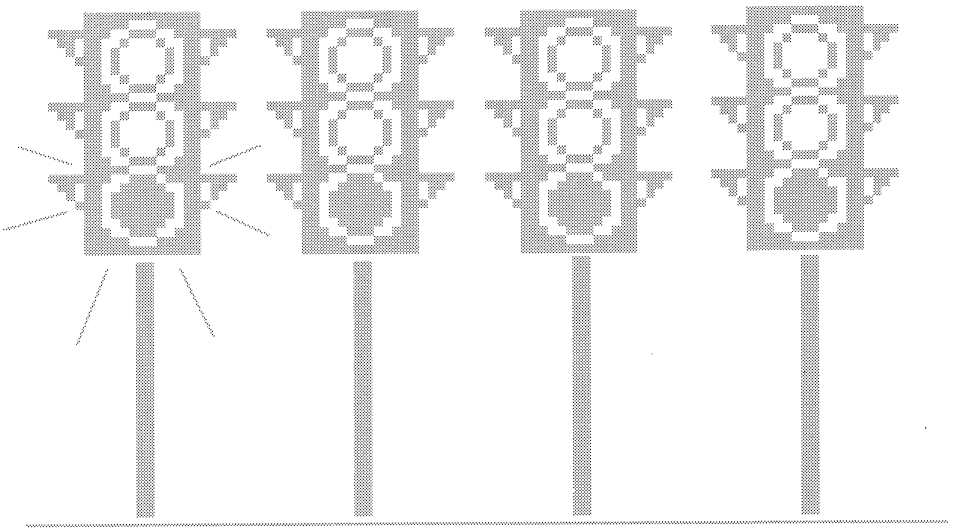
LOCK [#] *Dateinummer* [,{*Datensatz* | [*Beginn*] **TO** *Ende*}]
.
.
.
UNLOCK [#] *Dateinummer* [,{*Datensatz* | [*Beginn*] **TO** *Ende*}]

Dokumentationsbeurteilung

Wenn Sie Kommentare oder Anregungen bezüglich eines der zu diesem Produkt gehörenden Handbücher haben, oder Probleme mit der Software auftreten, so verwenden Sie bitte das Formblatt "Dokumentationsbeurteilung".

1. Teil: Erste Schritte

Q 1. Teil



Erste Schritte

Der 1. Teil, "Erste Schritte", führt den Benutzer in die Microsoft QuickBASIC-Umgebung ein und geht dann zu einer einleitenden Erläuterung der BASIC-Programmierung über. Jede grundlegende Eigenschaft der Umgebung wird zuerst erklärt und anschließend in einer kurzen Übung demonstriert. Jedes Kapitel endet mit einem Hinweis auf weitere Informationen zu den entsprechenden Themen.

Kapitel 1 erklärt die Installation von QuickBASIC anhand des gelieferten SETUP-Programms.

Kapitel 2 und 3 führen den Benutzer in die Arbeit mit den Kurzmenüs, Dialogfeldern und Fenstern ein.

Kapitel 4 erläutert die BASIC-Sprache für Programmieranfänger und sollte zuerst gründlich durchgearbeitet werden. Wenn Sie bereits in einer anderen Sprache programmiert haben, trägt dieses Kapitel zu der Vervollständigung der im 2. Teil, "Die Programmentwicklung in der QuickBASIC-Umgebung", erworbenen Kenntnisse bei, so daß Sie mit dem Schreiben eigener Programme beginnen können. Als Programmieranfänger empfiehlt es sich, einige der am Ende dieses Kapitels aufgelisteten Bücher über BASIC zu lesen.

Kapitelverzeichnis

1 Installieren von Microsoft QuickBASIC
2 Verwendung der Menüs und Befehle
3 Die QuickBASIC-Fenster
4 BASIC für Anfänger

1 Installieren von Microsoft QuickBASIC

Das Microsoft QuickBASIC-Paket enthält unter anderem die Dateien *setup.exe* und *setup.bat* (auf der 1. Diskette, Setup/QB-Express). Dieses als SETUP bezeichnete Programm erleichtert die Installation von QuickBASIC und erlaubt dem Benutzer die Wahl der zu installierenden Anzeigeoptionen oder die Aktualisierung der von QuickBASIC und der Hardware erforderten Dateien.

Dieses Kapitel behandelt:

- Die Installation von QuickBASIC.
- Die Verwendung des QB-Express.
- Wie Sie QuickBASIC – je nach Ihren Vorkenntnissen – am besten kennenlernen.

Das Programm SETUP

Wenn Sie keine Festplatte haben, vergewissern Sie sich, daß Sie über sechs formatierte, 360K-Disketten verfügen (oder über den entsprechenden Speicherplatz auf Disketten anderer Größe). Auf der Festplatte erfordert die Installation aller QuickBASIC-Dateien einen Speicherplatz von 1,8 Megabyte. Das SETUP-Programm läßt sich wie folgt starten:

◊ Legen Sie die 1. Diskette (Installation/QB-Express) ins Disketten-Laufwerk ein und geben Sie `setup` ein.

Sämtliche Anzeigen des SETUP-Programms enthalten folgende Elemente:

- Eine Erklärung der aktuellen oder der folgenden Anzeige.
- Das Installationsmenü, welches die Auswahl der nächsten Operation ermöglicht. Im Allgemeinen gibt das Installationsmenü jeder Anzeige dem Benutzer die Möglichkeit, die Tätigkeit fortzusetzen, zu einer vorhergehenden Anzeige zurückzukehren, Information zu den ausgewählten Optionen zu erhalten oder zu der Hauptanzeige zurückzukehren. Sie können sich im Menü mit Hilfe der RICHTUNGSTASTEN bewegen und eine Auswahl treffen, indem Sie die EINGABETASTE betätigen, wenn die gewünschte Auswahl hervorgehoben wird.

- Eine Eingabeaufforderung am unteren Ende der Anzeige gibt den nächsten Arbeitsschritt an.

Hinweis Wenn Sie beim Installieren von QuickBASIC anhand des SETUP-Programms einen Fehler machen, können Sie zu diesem Punkt zurückkehren und den Vorgang wiederaufnehmen. SETUP kann beliebig oft wiederholt werden.

Das Setup-Hauptmenü

Die einzelnen Punkte des Setup-Hauptmenüs ermöglichen die Auswahl der Vorgaben für alle Optionen (Kurzsetup) und das Anpassen der gewählten Optionen während der Installation (vollständiges Setup).

Für den Programmieranfänger ist das Kurzsetup zu empfehlen. Nachdem Sie QuickBASIC besser kennen, können Sie diese Optionen von der QuickBASIC-Umgebung aus erneut festlegen, ohne SETUP nochmals auszuführen.

Dem erfahrenen Programmierer gibt das vollständige Setup die Möglichkeit, den Bildschirm und das Verhalten von QuickBASIC den eigenen Bedürfnissen anzupassen. Kapitel 20, "Das Menü Optionen", zeigt, wie die während der Installation gewählten Suchpfade und Farben verändert werden können. (Sehen Sie unter den Befehlen **Bildschirm** und **Suchpfade festlegen** aus dem Menü **Optionen** nach.)

Verwenden Sie vor Abschluß der Installation den Befehl **Optionen anzeigen** sowohl bei der Wahl des vollständigen als auch des Kurzsetups.

Die Installation

Nach der Wahl der gewünschten Optionen, wählen Sie **Installation durchführen** aus dem Menü, um mit der Installation zu beginnen. Während der Installation fordert Sie QuickBASIC auf, die Disketten der Reihe nach in das Laufwerk einzulegen.

Auf einer Festplatte legt QuickBASIC die erforderlichen Dateien in den Verzeichnissen ab, die entweder im Kurzsetup oder im vollständigen Setup angegeben wurden. Falls die Verzeichnisse noch nicht existieren, legt sie QuickBASIC wie angegeben auf der Festplatte an.

Wenn Sie keine Festplatte haben, benötigen Sie zu Beginn der Installation fünf formatierte Disketten, die Sie dann nach Aufforderung von QuickBASIC in das Laufwerk einlegen. Es ist ratsam, die Disketten mit den ursprünglichen Namen und Nummern der Originaldisketten zu kennzeichnen.

QB-Express

Kehren Sie nach Abschluß der Installation zum Setup-Hauptmenü zurück. Auch wenn Ihnen QuickBASIC bekannt ist, sollten Sie die Befehle **Ende** und **QB-Express** vom Hauptmenü wählen. Microsoft QB-Express, ein kurzes Lernprogramm, führt Sie in die Programmierumgebung ein und zeigt die Arbeitsweise von QuickBASIC. Wenn Sie mit QuickBASIC bereits vertraut sind, macht Sie QB-Express auf die Verbesserungen in der Programmierumgebung der Version 4.5 aufmerksam.

Programmieren mit QuickBASIC

QuickBASIC ist zwar eine hervorragende Umgebung, um das Programmieren zu erlernen, es wurde jedoch nicht speziell als "Programmierlernumgebung" entwickelt. Außer einer leistungsfähigen Version der BASIC-Sprache stellt QuickBASIC alle zum Programmieren erforderlichen Hilfen – Bearbeitung, Fehlerbeseitigung und On-Line-Hilfe – zur Verfügung.

Kapitel 2 bis 9 dieses Handbuches bilden das Lehrmaterial zu QuickBASIC, und das links stehende Schreibmaschinensymbol kennzeichnet alle praktischen Übungen.

Für den Programmieranfänger

Beim Erlernen von QuickBASIC müssen Sie zwischen den von der Umgebung zur Verfügung gestellten Hilfen und der Programmiersprache selbst unterscheiden. Kapitel 2, "Verwendung der Menüs und Befehle", und Kapitel 3, "Die QuickBASIC-Fenster", bieten dem Benutzer die beste Möglichkeit, die Umgebung kennenzulernen. Anschließend ist Kapitel 4, "BASIC für Anfänger", zu empfehlen, um Ihr Wissen zur BASIC-Programmierung soweit zu vervollständigen, damit Sie mit dem 2. Teil, "Die Programmentwicklung in der QuickBASIC-Umgebung", beginnen können.

Der 2. Teil, "Die Programmentwicklung in der QuickBASIC-Umgebung", verdeutlicht die Bearbeitung, Fehlerbeseitigung und Verwendung der On-Line-Hilfe anhand eines erweiterten Programmbeispiels. Sie werden die Gelegenheit haben, den Code eines reellen Anwendungsprogramms, *qkarten.bas*, in mehreren Übungen zu ergänzen. Jedes Kapitel erfordert ein bis zwei Arbeitsstunden.

Obwohl "Die Programmentwicklung in der QuickBASIC-Umgebung" kein Lehrbuch zum Thema Programmieren darstellt, ist *qkarten.bas* ein gutes Beispiel für strukturiertes Programmieren. Der Code in *qkarten.bas* ist auch für andere Programme anwendbar, das Hauptziel von *qkarten.bas* bleibt jedoch die schnelle Einführung in die QuickBASIC-Umgebung.

Sie brauchen nicht alle Programmiertechniken zu verstehen, bevor Sie den Code für *qkarten.bas* eingeben. Um das Programmieren eingehender zu lernen, empfiehlt es sich, einige der am Ende des 4. Kapitels, "BASIC für Anfänger", aufgeführten Programmlistings zu lesen. Das in diesem Paket mitgelieferte Handbuch *Programmieren in BASIC* enthält ausführliche Beispiele und Erklärungen ausgewählter Programmierthemen.

Für den fortgeschrittenen BASIC-Programmierer

Wenn Ihnen BASIC bereits geläufig ist, werden Sie in der Lage sein, Kapitel 2, "Verwendung der Menüs und Befehle", und Kapitel 3, "Die QuickBASIC-Fenster", die Grundkenntnisse zu Menüs und Fenstern, sowie zum Öffnen, Löschen und Speichern von Dateien vermitteln, schnell durchzugehen. Kapitel 4 können Sie in diesem Fall überblättern und direkt zu Teil 2, "Die Programmentwicklung in der QuickBASIC-Umgebung", übergehen, um die weiterentwickelten Eigenschaften von QuickBASIC, wie Bearbeitung, On-Line-Hilfe und Fehlerbeseitigung kennenzulernen. Selbst wenn Sie bereits mit früheren Versionen von QuickBASIC gearbeitet haben, sind neue Programmiertechniken in *qkarten.bas* zu finden. Dabei handelt es sich um ein Beispielprogramm, das vervollständigt wird, während Sie sich mit den erweiterten Eigenschaften der Umgebung vertraut machen. Falls die dazugehörigen Kommentare nicht ausreichen, werden die Programmiertechniken außerdem in dem Handbuch *Programmieren in BASIC* beschrieben.

Die Arbeit mit der Maus

QuickBASIC unterstützt zu der Microsoft-Mouse kompatible Zeigegeräte. Eingehende Informationen über den Einsatz von QuickBASIC mit der Maus sind Abschnitt 10.7 zu entnehmen.

2 Verwendung der Menüs und Befehle

Dieses Kapitel erklärt das Starten von Microsoft QuickBASIC und die Verwendung der Kurzmenüs. Ein "Menü" stellt eine Liste von Befehlen dar. QuickBASIC bietet die Auswahl zwischen zwei Menüsätzen, dem vollständigen und dem Kurzmenü. Das Kurzmenü, welches in diesem Abschnitt zur Anwendung kommt, enthält alle zum Einsatz von QuickBASIC erforderlichen Befehle für einen Programmieranfänger. Im 3. Teil, "QuickBASIC-Menüs und -Befehle", werden die weiteren Befehle des vollständigen Menüs beschrieben.

Nach Abschluß dieses Kapitels werden Sie in der Lage sein:

- Menüs und Dialogfelder zu öffnen und zu verwenden.
- Fragen anhand der On-Line-Hilfe von QuickBASIC zu beantworten.
- Funktionstasten und Tastenkombinationen zur direkten Befehlseingabe zu benutzen.

Dieses Kapitel erfordert ein bis anderthalb Arbeitsstunden.

Starten von QuickBASIC

QuickBASIC läßt sich wie folgt starten:

1. Geben Sie `qb` nach der DOS-Anfrage ein.
2. Betätigen Sie die EINGABETASTE, um eine Einführung in die QuickBASIC-Umgebung zu erhalten, und gehen Sie den Text mit Hilfe der Tasten BILD↑ und BILD↓ durch.
3. Löschen Sie dieses Informationsfenster durch Betätigen der ESC-TASTE vom Bildschirm.

Hinweis Wenn der Computer über einen einfarbigen oder Flüssigkristall-Bildschirm verfügt, oder Sie einen Color Graphics Adapter (CGA) mit einem schwarz-weiß Bildschirm verwenden, sind zusätzliche QB-Aufrufoptionen den Abschnitten 10.1.1., "Der QB-Befehl", und 10.1.2, "Die Option /NOHI", zu entnehmen.

Beim Starten von QuickBASIC mit Hilfe der Option /NOHI oder /B/NOHI kann es vorkommen, daß die in diesem Handbuch als hervorgehoben dargestellten Buchstaben unterschiedlich aussehen.

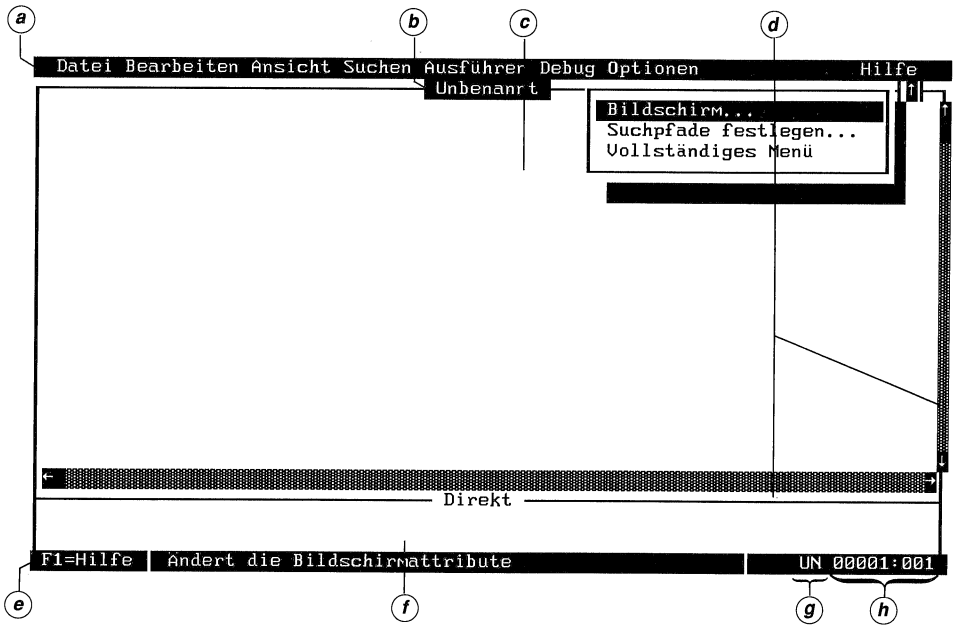
Die QuickBASIC-Umgebung

Die QuickBASIC-Programmierwerkzeuge stehen Ihnen jederzeit zur Verfügung. Sie bieten vielseitige Eigenschaften für die Programmorganisation, Bearbeitung, Syntaxüberprüfung, Dateiverwaltung, Ausgabe und Fehlerbeseitigung. Sie können sogar die Umgebungsfarben anpassen.

Die Menüzeile

Die in der Menüzeile am oberen Ende der QuickBASIC-Anzeige aufgeführten Menüs enthalten die QuickBASIC-Umgebungsbefehle (siehe Abbildung 2.1), die beim Schreiben und Ändern des Programms im Arbeitsfenster benutzt werden.

Abbildung 2.1 QuickBASIC-Aufrufanzeige



a) Menüzeile.
b) Titelleiste.
c) Arbeitsfenster.
d) Bildlaufleisten.
e) Referenzzeile.
f) Direktes Fenster.
g) Anzeige der UMSCHALT- und NUM-FESTSTELLTASTE.
h) Anzeige von Zeile und Spalte.

Die Referenzzeile

Die Referenzzeile am unteren Ende des Bildschirms gibt die für wichtige Aktionen zu betätigenden Tasten an (siehe Abbildung 2.1). Während der Arbeit an dem Programm aktualisiert die Referenzzeile die Umgebungs- und Programminformationen. Der hervorgehobene Befehl wird jeweils durch eine Kurzbeschreibung in der Referenzzeile ergänzt. Die Anzeige von Zeile und Spalte gibt die Position im Programm und die Anzeige der UMSCHALT- und NUM-FESTSTELLTASTEN die jeweilige Einstellung der Tastatur an.

Das Öffnen der Menüs

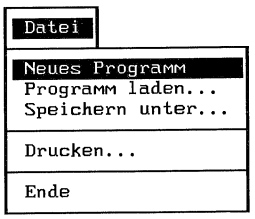
Die meisten QuickBASIC-Befehle lassen sich durch deren Auswahl in den Menüs ausführen. Wenn die ALT-TASTE betätigt wird, hebt QuickBASIC das Dateimenü und einen der Buchstaben der Menübezeichnungen hervor. Zum Öffnen eines Menüs betätigen Sie zuerst die ALT-TASTE und drücken anschließend die dem hervorgehobenen Buchstaben der Menübezeichnung entsprechende Taste. Sie können auch die Richtungstasten (die Pfeiltasten NACH OBEN, NACH UNTEN, NACH LINKS und NACH RECHTS) benutzen, um die Menüs zu öffnen und sich von einem Menü zum anderen zu bewegen.

Ein Menü läßt sich wie folgt öffnen:

1. Wählen Sie das Dateimenü durch Betätigen der ALT-TASTE aus.

   Beim "Auswählen" (Hervorheben) eines Menüs, wird dessen Bezeichnung invers dargestellt. Bei nochmaligem Betätigen der ALT-TASTE wird die Menüzeile wieder normal dargestellt.
2. Öffnen Sie das Menü **Datei** durch Drücken der EINGABETASTE (siehe Abbildung 2.2).

Abbildung 2.2 Das Menü Datei



- Wählen Sie aufeinanderfolgende Befehle eines geöffneten Menüs durch Betätigen der NACH OBEN oder NACH UNTEN TASTE aus.
- Öffnen Sie benachbarte Menüs durch Drücken der NACH RECHTS, oder NACH LINKS TASTE.

3. Drücken Sie die ESC-TASTE, um ein Menü zu schließen, ohne einen Befehl auszuführen. Der Cursor kehrt zum Arbeitsfenster zurück, dem Bildschirmteil, in dem das Programm bearbeitet wird. Die ESC-TASTE ermöglicht es dem Benutzer, jede Situation jederzeit rückgängig zu machen.

Die On-Line-Hilfe von QuickBASIC

Zusätzliche Informationen über die Menüs oder andere QuickBASIC-Eigenschaften, über BASIC-Schlüsselwörter, Sprachelemente, die QuickBASIC-Umgebung und Fehlermeldungen sind in der On-Line-Hilfe zu finden.

Verwendung der On-Line-Hilfe

Es gibt mehrere Möglichkeiten, die On-Line-Hilfe von QuickBASIC anzufordern. Für Fragen bezüglich der Menübezeichnungen oder Befehle, wählen Sie die entsprechende Bezeichnung aus, und betätigen Sie dann F1. Falls Sie in einem bestimmten Programmteil Hilfe benötigen, positionieren Sie den Cursor innerhalb des Wortes, und drücken Sie F1.

Innerhalb der On-Line-Hilfe können Sie sich wie folgt bewegen:

- Drücken Sie BILD↓, um die Anzeige bildschirmweise durchzugehen.
- Drücken Sie BILD↑, wenn Sie vorhergehende Anzeigen erneut sehen wollen.
- Verlassen Sie die On-Line-Hilfe durch Betätigen der ESC-TASTE.

Für eine allgemeine Beschreibung der On-Line-Hilfsfunktion ist F1 bei gedrückter UMSCHALTTASTE zu betätigen.

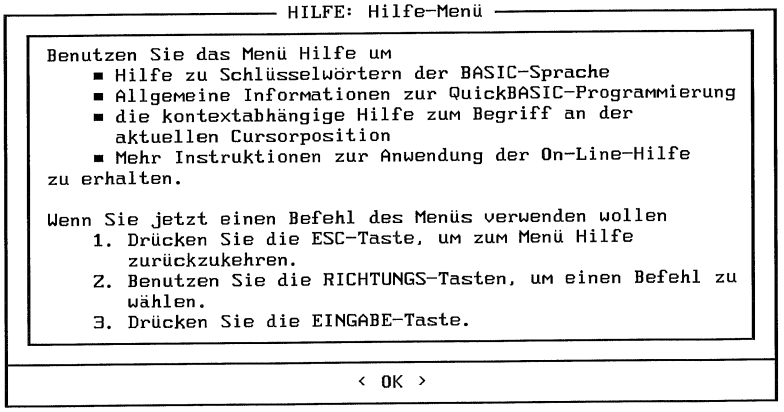
Das Hilfsmenü

Die On-Line-Hilfe ist größtenteils über das Hilfsmenü verfügbar und kann wie folgt verwendet werden:

1. Drücken Sie die ALT-TASTE, um das Menü **Datei** auszuwählen, ohne es zu öffnen.
2. Wählen Sie das Menü **Hilfe** durch Betätigen der NACH LINKS TASTE aus.

3. Wenn Sie jetzt F1 betätigen, zeigt die On-Line-Hilfe die in Abbildung 2.3 dargestellten Informationen über das Hilfsmenü an.

Abbildung 2.3 Dialogfeld des Hilfsmenüs



4. Drücken Sie die EINGABETASTE, nachdem Sie den Text durchgelesen haben.
5. Öffnen Sie das Hilfsmenü durch Betätigen der NACH UNTEN TASTE.

Wenn Sie Näheres über ein Element des Hilfsmenüs wissen möchten, heben Sie den gewünschten Befehl mit Hilfe der RICHTUNGSTASTEN hervor, und betätigen Sie F1, um weitere Erklärungen zu erhalten.

Wenn es für das aktuelle Element keine On-Line-Hilfe gibt, oder wenn die betreffende Hilfsdatei nicht verfügbar ist, gibt QuickBASIC eine Meldung, einen Piepton oder beides aus.

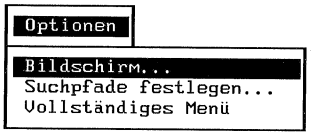
Das Hilfsmenü besteht aus folgenden Befehlen:

Befehl	***Funktion***
Index	Zeigt eine alphabetische Liste aller QuickBASIC-Schlüsselwörter an. Positionieren Sie den Cursor auf ein Schlüsselwort, und betätigen Sie F1, um Hilfe für das jeweilige Schlüsselwort zu erhalten.
Themen	Zeigt eine Übersicht des Inhalts der On-Line-Hilfe. Ausführliche Informationen über jede aufgeführte Kategorie erhalten Sie durch Positionieren des Cursors auf die Bezeichnung der betreffenden Kategorie und Drücken der Taste F1.
Begriff	Liefert eine Hilfe für jedes an der Cursorstelle stehende Wort auf dem Bildschirm. Ein auf der Cursorstelle stehendes Wort erscheint im Menü **Hilfe** nach dem Befehl **Begriff**. Ist kein Wort an der Cursorstelle vorhanden, bleibt der Befehl **Begriff** inaktiv.
Überblick	Erläutert die Verwendung der On-Line-Hilfe.

Die Befehlsauswahl

Um einen Befehl "auszusuchen" (auszuführen), wählen Sie ihn durch Betätigen der NACH OBEN- oder NACH UNTEN TASTE aus, und drücken Sie danach die EINGABETASTE oder den hervorgehobenen Buchstaben der Befehlsbezeichnung. (Ist in einer Befehlsbezeichnung kein Buchstabe hervorgehoben, kann der entsprechende Befehl nicht verwendet werden.)

Wenn einem Befehl drei Punkte folgen (...), erscheint ein Dialogfeld (Beschreibung folgt) bei Auswahl des Befehls. Die nicht von drei Punkten gefolgten Befehle werden sofort nach deren Auswahl ausgeführt.



Um einen Befehl vor der Auswahl besser kennenzulernen, sehen Sie sich die On-Line-Hilfe durch Betätigen der Taste F1 an. Löschen Sie die Hilfsinformation nach dem Durchlesen mit der ESC-TASTE, und drücken Sie die EINGABETASTE, wenn Sie den Befehl ausführen möchten. Führen Sie folgende Arbeitsschritte durch:

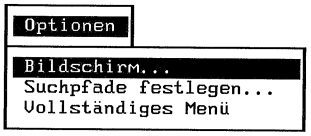
1. Drücken Sie die ALT-TASTE und dann O, um das Menü **Optionen** auszuwählen.

   Diese Tastenkombination öffnet das Menü **Optionen**. In diesem Handbuch werden Tastenkombinationen durch die jeweiligen mit einem Pluszeichen verbundenen Tastenbezeichnungen angegeben. Demnach ist die oben erwähnte Kombination als ALT+O zu sehen.
2. Halten Sie die NACH UNTEN TASTE nieder, bis der Befehl **vollständiges Menü** hervorgehoben wird.
3. Drücken Sie F1, um die On-Line-Hilfe erscheinen zu lassen, während der Befehl **vollständiges Menü** hervorgehoben bleibt.
4. Löschen Sie die On-Line-Hilfe durch Betätigen der ESC-TASTE.
5. Schalten Sie die vollständigen Menüs mit der EINGABETASTE aus, falls der Befehl durch einen Punkt (•) gekennzeichet ist. Betätigen Sie die ESC-TASTE, um das Menü **Optionen** zu schließen, ohne dessen Befehle auszuführen, wenn der Befehl **vollständiges Menü** nicht durch einen Punkt gekennzeichnet ist.

   Der Befehl **vollständiges Menü** stellt einen "Umschalter" dar. Wenn er mit einem Punkt versehen ist, wird das vollständige Menü bei Auswahl ausgeschaltet, ist kein Punkt vorhanden, wird es bei Auswahl eingeschaltet.
6. Öffnen Sie erneut das Menü **Optionen** über die Tastenkombination ALT+O. Vergewissern Sie sich, daß der Befehl **vollständiges Menü** nicht mit einem Punkt versehen ist, und schließen Sie das Menü durch Betätigen der ESC-TASTE.

Die Betätigung der RICHTUNGSTASTEN ist nur beim Anfordern von Hilfe für einen Befehl notwendig. Wenn Sie keine Hilfe benötigen, brauchen Sie lediglich die jeweiligen Befehle durch Drücken der Buchstabentasten auszuwählen, die den hervorgehobenen Buchstaben der Befehlsbezeichnungen entsprechen.

Verwendung der Dialogfelder



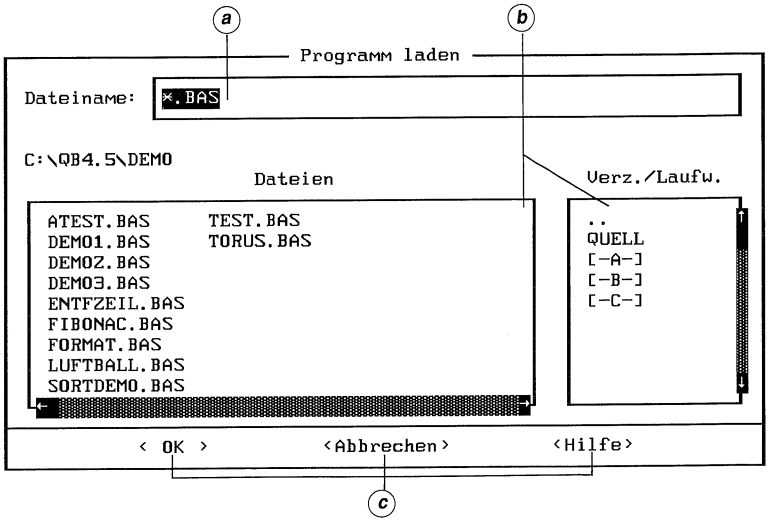
Ein "Dialogfeld" erscheint bei der Auswahl eines beliebigen, von drei Punkten (...) gefolgten Menübefehls. QuickBASIC fordert den Benutzer zur Eingabe der zur Ausführung eines Befehls nötigen Informationen auf. Dialogfelder erscheinen auch in folgenden Situationen:

- Bei Eingabe einer Codezeile, die einen Syntaxfehler enthält.
- Während der Ausführung eines Programms, das einen Syntaxfehler enthält.
- Bei Auswahl der Befehlsfläche **Hilfe** in einem Dialogfeld.
- Bei Hilfeanforderung für ein Menü oder einen Menübefehl.
- Beim Versuch, eine Arbeitssitzung zu beenden, ohne die Daten zu speichern.

Zusammensetzung eines Dialogfeldes

Dialogfelder verfügen über mehrere Einrichtungen, um QuickBASIC Informationen mitteilen zu können (siehe Abbildung 2.4).

Abbildung 2.4 Das Dialogfeld Programm laden



a) *Textfeld.*
b) *Verzeichnisfelder.*
c) *Befehlsflächen.*

Die folgende Aufstellung erklärt die Verwendung dieser Einrichtungen:

Einrichtung	*Beschreibung*
Textfeld	Geben Sie Textpassagen, wie zum Beispiel einen Dateinamen, in diesem Feld ein.
Verzeichnisfeld	Wählen Sie ein Element aus der Gruppe der aufgelisteten Elemente aus.
Befehlsfläche	Anhand dieser Einrichtung können Sie Eingaben bestätigen, Hilfe anfordern und das Dialogfeld verlassen.

Anzeige eines Dialogfeldes

Wenn es zum ersten Mal geöffnet wird, enthält das Dialogfeld "Standardeinstellungen", die die gebräuchlichsten Einstellungen darstellen. Danach zeigt es die bei dessen letzter Anzeige getroffene Auswahl an. Die eckigen Klammern der Standardbefehlsfläche werden hervorgehoben. Sie können die Standardbefehlsfläche anhand der EINGABETASTE auswählen oder sich im Dialogfeld mit Hilfe von ALT-Tastenkombinationen oder TAB bewegen.

Um das Dialogfeld **Programm laden** aus dem Menü **Datei** zu öffnen und anzusehen, verfahren Sie wie folgt:

1. Falls das Menü **Optionen** noch geöffnet ist, schließen Sie es durch Betätigen der ESC-TASTE.
2. Drücken Sie die Tastenkombination ALT+D und die Taste P, um den Befehl **Programm laden** aus dem Menü **Datei** auszuwählen.

   Anschließend erscheint das in Abbildung 2.4 dargestellte Dialogfeld **Programm laden**, das das Laden von Programmdateien in das Arbeitsfenster ermöglicht.
3. Rufen Sie die Hilfe über die Taste F1 auf.

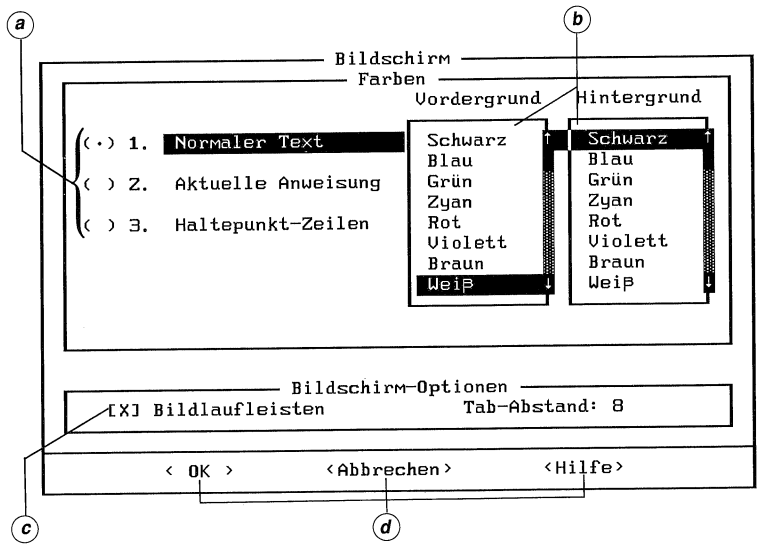
   Ein Hilfsdialogfeld erscheint am oberen Ende des Dialogfeldes **Programm laden**. Sie könnnen sich im Hilfstext mit Hilfe der RICHTUNGSTASTEN bewegen.
4. Betätigen Sie die ESC-TASTE, um das Hilfsdialogfeld zu löschen, und den Cursor in das Dialogfeld **Programm laden** zurückzubringen.
5. Drücken Sie die ALT-TASTE, um die Buchstaben hervorzuheben, die zusammen mit der ALT-TASTE betätigt werden können.
6. Bewegen Sie den Cursor durch Betätigen der TAB-TASTE in das Verzeichnisfeld **Dateien**.

   Wenn das Verzeichnisfeld zahlreiche Elemente enthält, können Sie sich durch Betätigen der RICHTUNGSTASTEN oder des ersten Buchstabens des gewünschten Elements innerhalb des Feldes bewegen.
7. Drücken Sie die TAB-TASTE oder ALT+V, um den Cursor in das Verzeichnisfeld **Verz./Laufw.** zu bringen.
8. Schließen Sie das Dialogfeld mit der ESC-TASTE, ohne ein Programm zu öffnen.

Andere Einrichtungen eines Dialogfeldes

Andere Einrichtungen eines Dialogfeldes umfassen Optionsfelder und Kontrollfelder. Mit Hilfe dieser Einrichtungen können bei der Installation von QuickBASIC gewählte Optionen, wie Bildschirmfarben, Bildlaufleisten und der Abstand zwischen Tabulatorstops im Dialogfeld **Bildschirm** aus dem Menü **Optionen** geändert werden (siehe Abbildung 2.5).

Abbildung 2.5 Das Dialogfeld Bildschirm



a) Optionsflächen, die anhand der NACH OBEN TASTE und NACH UNTEN TASTE ausgewählt werden können.

b) Auswahlfelder, aus denen ein Element auszuwählen ist.

c) Kontrollfeld, betätigen Sie die LEERTASTE zur Auswahl.

d) Befehlsflächen, mit deren Hilfe Einstellungen bestätigt, Hilfe angefordert oder das Menü verlassen werden kann.

Optionsflächen sind aktiviert, wenn sie mit einem in Klammern stehenden Punkt (•) versehen sind. Es können nicht mehrere Optionen in einer Gruppe von Optionsflächen gleichzeitig aktiviert werden. Ein in Klammern stehendes X deutet darauf hin, daß das Kontrollfeld aktiviert ist. Es kann eine beliebige Anzahl von Kontrollfeldern gleichzeitig aktiviert werden.

Optionen lassen sich wie folgt einstellen:

1. Drücken Sie ALT+O und anschließend ALT+B, um den Befehl **Bildschirm** aus dem Menü **Optionen** auszuwählen und das in Abbildung 2.5 dargestellte Dialogfeld **Bildschirm** erscheinen zu lassen.
2. Bringen Sie den Cursor über die Tastenkombination ALT+T in das Textfeld Tab-Abstand.

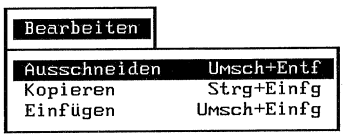
   Geben Sie die Anzahl der Leerstellen zwischen den Tabulatorstopps neu ein, wenn Sie diesen Abstand während der Bearbeitung ändern möchten. Vier Leerstellen bilden einen für die Programmierung geeigneten Abstand.
3. Bewegen Sie den Cursor durch Betätigen der Tastenkombination ALT+L in das Kontrollfeld **Bildlaufleisten**. Drücken Sie die Leertaste, um ein `X` in ein leeres Kontrollfeld einzusetzen oder es zu entfernen, falls das Kontrollfeld bereits damit versehen ist. (Wie aus Abbildung 2.1 ersichtlich, sind Bildlaufleisten nur zusammen mit einer Maus verwendbar.)
4. Die drei Typen von Vorder- und Hintergrundfarben lassen sich wie folgt festlegen:
   a. Drücken Sie die ALT-TASTE und die hervorgehobene Zahl (`1`, `2`, oder `3`), um die Option auszuwählen, deren Farben Sie ändern wollen.
   b. Drücken Sie die TAB-TASTE, um zu den Feldern **Vordergrund** oder **Hintergrund** zu gelangen, und wählen Sie die Farben mit Hilfe der NACH OBEN TASTE und NACH UNTEN TASTE aus.
   c. Wiederholen Sie die zwei vorhergehenden Arbeitsschritte, um die Optionen **Aktuelle Anweisung** und **Haltepunkt-Zeilen** in gleicher Weise zu ändern.
5. Betätigen Sie die EINGABETASTE, um die getroffene Auswahl zu bestätigen, oder drücken Sie die ESC-TASTE, um die ursprünglichen, vor der Anzeige des Dialogfeldes bestehenden Einstellungen wiederherzustellen.

Die Option **Aktuelle Anweisung** bestimmt die Attribute der im Programmablauf gegenwärtig ausgeführten Zeile. Die Option **Haltepunkt-Zeilen** setzt die Farben der mit Haltepunkten versehenen Zeilen fest. Es ist zu beachten, daß die Farben der **Haltepunkt-Zeilen** und der Option **Aktuelle Anweisung** nur während der Programmausführung zu sehen sind.

Die direkte Befehlseingabe

Befehle können mit Hilfe spezifischer Tasten und Tastenkombinationen direkt eingegeben werden, ohne Menüs zu öffnen und deren Elemente auszuwählen. Für zahlreiche QuickBASIC-Aufgaben stehen Tasten zur direkten Befehlseingabe zu Ihrer Verfügung.

Tasten zur direkten Befehlseingabe



Wenn ein Befehl über eine Taste zur direkten Befehlseingabe verfügt, steht diese neben dem Befehl. Anstatt beispielsweise das Menü **Bearbeiten** zu öffnen und die Befehle **Ausschneiden**, **Kopieren** oder **Einfügen** auszuwählen, können Sie die Tastenkombinationen UMSCHALT+ENTF, STRG+EINFG und UMSCHALT+EINFG betätigen.

Weitere Tastenkombinationen

Es gibt auch weitere Sondertasten und Tastenkombinationen. Zum Beispiel bringt UMSCHALT+F1 den Überblick auf den Bildschirm, und F1 läßt die On-Line-Hilfe für Menübezeichnungen und Befehle oder für jegliche an der Cursorstelle stehenden Schlüsselwörter oder Programmsymbole erscheinen. Andere Tastenkombinationen verwenden Funktionstasten, zum numerischen Tastenblock gehörende Tasten wie EINFG und ENTF und zahlreiche Kombinationen der UMSCHALT-, ALT-, oder STRG-TASTEN mit anderen Tasten.

Der Cursor kann zum Beispiel mit Hilfe der NACH RECHTS und NACH LINKS TASTEN nach rechts und nach links auf dem Bildschirm bewegt werden. Um den Cursor zum nächsten Wort nach der aktuellen Cursorposition zu bringen, ist die NACH LINKS oder NACH RECHTS TASTE bei niedergedrückter STRG-TASTE zu betätigen.

Die nächste Übung verdeutlicht die Arbeitsweise der gebräuchlichsten Tasten und Tastenkombinationen:

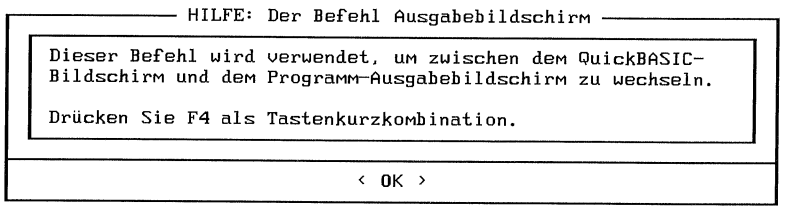
1. Öffnen Sie das Menü **Ansicht** über ALT+N.

   Der Befehl **SUBs** wird automatisch ausgewählt. Beachten Sie die rechts von der Befehlsbezeichnung stehende Taste zur direkten Befehlseingabe (F2).

2. Betätigen Sie die NACH UNTEN TASTE, um den Befehl **Ausgabebildschirm** auszuwählen, der mit F4 als Taste zur direkten Befehlseingabe versehen ist.

3. Drücken Sie F1, um die On-Line-Hilfe zu erhalten (siehe Abbildung 2.6).

Abbildung 2.6 Hilfe für den Befehl Ausgabebildschirm



4. Schließen Sie die Hilfsanzeige durch Betätigen der EINGABETASTE, die die Taste zur direkten Befehlseingabe für die Standardbefehlsfläche darstellt (oder für die ausgewählte Befehlsfläche).
5. Schließen Sie das Menü **Ansicht** mit Hilfe der ESC-TASTE.

Verlassen von QuickBASIC

Am Ende einer Arbeitssitzung prüft QuickBASIC, ob alle aktuellen Daten gespeichert wurden. Wenn das nicht der Fall ist, fragt ein Dialogfeld den Benutzer, ob diese Daten gespeichert werden sollen. Da im Laufe dieser Arbeitssitzung kein Text eingegeben wurde, drücken Sie N, nachdem Sie das Dialogfeld wie folgt auf den Bildschirm gebracht haben:

◊ Drücken Sie ALT+D, dann E, um den Befehl **Ende** aus dem Menü **Datei** auszuwählen und zu DOS zurückzukehren.

Im Dialogfeld **Bildschirm** vorgenommene und bestätigte Änderungen werden beim Verlassen von QuickBASIC in einer Datei mit dem Namen *qb.ini* gespeichert und beim erneuten Starten von QuickBASIC als Standard benutzt.

Weitere Informationen

Weitere Informationen zu den in diesem Kapitel erläuterten Themen sind folgenden Abschnitten zu entnehmen:

Abschnitte	*Informationen*
Abschnitt 10.1.1., "Der QB-Befehl", und 10.1.2, "Die Option /NOHI"	Erklärung der QuickBASIC Aufrufoptionen
Abschnitt 10.8, "Verwendung der On-Line-Hilfe", und Kapitel 21, "Das Hilfsmenü"	Vollständige Beschreibung der On-Line-Hilfe
Abschnitt 10.2, "Öffnen der Menüs und Auswahl der Befehle" und 10.3, "Verwendung der Dialogfelder"	Weitere Informationen über Menüs und Dialogfelder
Abschnitt 10.2.2., "Verwendung der Tasten zur direkten Befehlseingabe"	Eine Aufstellung der am häufigsten verwendeten Tasten zur direkten Befehlseingabe
Abschnitt 12.7, "Zusammenfassung der Bearbeitungsbefehle"	Eine vollständige Liste der zur Bearbeitung verwendeten Tasten-kombinationen

3 Die QuickBASIC-Fenster

Dieses Kapitel beschreibt den Einsatz der QuickBASIC-Fenster beim Schreiben eines kurzen Programms und stellt folgende Verfahren vor:

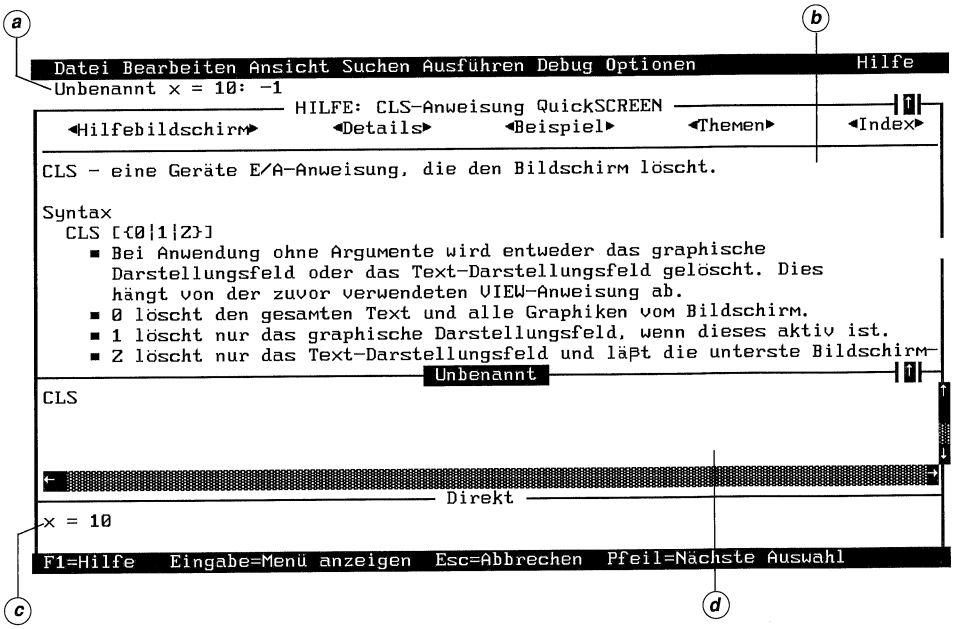
- Öffnen eines vorhandenen Programms und Bewegen des Cursors.
- Aktivieren, Bewegen und Ändern der Fenstergröße.
- Ausführen von BASIC-Anweisungen aus dem Direktfenster.
- Überwachen der Variablenwerte mit Hilfe des Debug-Fensters.
- Verwenden des Hilfsfensters.
- Speichern von Programmen als Dateien auf einer Diskette/Festplatte.

Dieses Kapitel erfordert etwa ein bis anderthalb Arbeitsstunden.

In den Kurzmenüs verfügbare Fenster

QuickBASIC wird als "fensterorientierte" Umgebung bezeichnet, da der Bildschirm in separate Bereiche aufgeteilt ist, in denen unterschiedliche Aufgaben durchgeführt werden können. Einige Fenster, wie zum Beispiel das Arbeits- oder Hilfsfenster, zeigen aus Dateien geladenes Material an. Andere Fenster, wie das Debugfenster, zeigen die Information von QuickBASIC zum aktuellen Programm an. In den Kurzmenüs sind die in Abbildung 3.1 gezeigten Fenster verfügbar.

Abbildung 3.1 Die in den Kurzmenüs verfügbaren Fenster



a) *Debugfenster.*
b) *Hilfsfenster.*
c) *Direktfenster.*
d) *Arbeitsfenster.*

Fenster-bezeichnung	***Beschreibung***
Arbeitsfenster	Das Fenster, in dem der größte Teil der Programmierarbeit durchgeführt wird. Beim Starten von QuickBASIC stellt das Arbeitsfenster das "aktive Fenster" dar, in dem der Cursor steht. Beim Laden einer Datei erscheint deren Inhalt im Arbeitsfenster.
Direktfenster	Das Fenster am unteren Ende des Bildschirms, das einzeilige BASIC-Codes zum Überprüfen von Programmierideen ausführt und in dem unterschiedliche Berechnungen durchgeführt werden können. Das Direktfenster läßt direkte Eingaben, jedoch nicht das Laden von Dateien zu.
Hilfsfenster	Ein Fenster, das durch Betätigen der F1-TASTE oder der Tastenkombination UMSCHALT+F1 am oberen Ende des Bildschirms geöffnet wird. Im Hilfsfenster werden Informationen über QuickBASIC-Schlüsselwörter, im Programm definierte Symbole und Hilfsanforderung angezeigt.
Debug-Fenster	Ein Fenster, das beim Auswählen bestimmter Befehle aus dem Debug-Menü am oberen Ende des Bildschirms erscheint.

Es können mehrere Fenster gleichzeitig geöffnet werden, wobei jedoch nur eines aktiv sein kann. Das Öffnen mehrerer Fenster ermöglicht dem Benutzer, auf unterschiedliche Informationen gleichzeitig zuzugreifen. Das Arbeitsfenster kann beispielsweise Programmcode enthalten, während die angeschlossene Hilfe für die gewünschte Anweisung angezeigt wird.

Das Arbeitsfenster

Im Arbeitsfenster wird der größte Teil der Programmierarbeit durchgeführt. Dieser Abschnitt erklärt, wie ein bestehendes Programm aus einer Datei in das Arbeitsfenster geladen und der Cursor in dessen Text bewegt wird. Anschließend haben Sie die Gelegenheit, dieses Programm aus dem Fenster zu löschen und eigene Codezeilen zu schreiben und auszuführen.

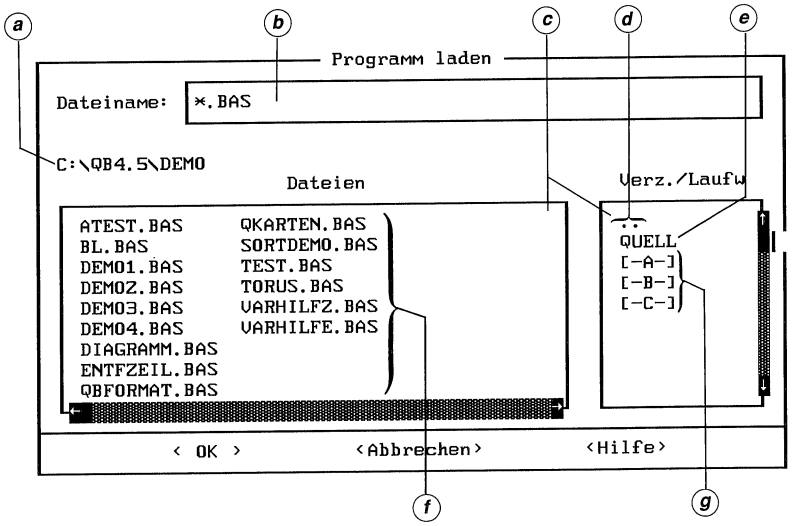
Das Laden von Programmen in das Arbeitsfenster

Beim Laden einer Datei von Diskette erscheint deren Text im Arbeitsfenster. Ein Programm läßt sich wie folgt laden:

1. Starten Sie QuickBASIC mit Hilfe des im 2. Kapitel erlernten Befehls QB.
2. Löschen Sie das Dialogfeld **Willkommen** durch Betätigen der ESC-TASTE.
3. Drücken Sie ALT+D und P, um den Befehl **Programm laden** aus dem Menü **Datei** auszuwählen.

   QuickBASIC zeigt ein Dialogfeld wie in Abbildung 3.2 an.

Abbildung 3.2 Das Dialogfeld Programm laden



a) *Pfadangabe, gibt den Pfad an, der zu den im Verzeichnisfeld Dateien aufgeführten Dateien führt.*
b) *Textfeld.*
c) *Verzeichnisfelder.*
d) *Symbol des nächsthöheren Verzeichnisses.*
e) *Bezeichnung des nächtsniederen Verzeichnisses.*
f) *Das Verzeichnisfeld enthält die Dateibezeichnungen, deren Erweiterung im Textfeld aufgeführt ist.*
g) *Verfügbare Laufwerke.*

Der Cursor befindet sich nun im Textfeld, in dem **.bas* ausgewählt ist. Eingegebene Zeichen ersetzen **.bas*, solange es ausgewählt bleibt. Die in diesem Feld stehende Zeichenfolge **.bas* weist QuickBASIC an, sämtliche Dateien mit dieser Erweiterung im Dialogfeld **Dateien** aufzulisten.

Dieses Dialogfeld weist folgende Einrichtungen auf:

Einrichtung	*Funktion*
Textfeld	Für Texteingabe, wie z.B. Dateibezeichnungen.
Verzeichnisfeld Dateien	Ein Feld, das Dateien des in der Pfadangabe angegebenen Verzeichnisses anzeigt.
Verzeichnisfeld Verz./Laufw.	Ein Feld, das die Verzeichnisse zeigt, die sich oberhalb und unterhalb des in der Pfadangabe angegebenen Verzeichnisses befinden sowie die im jeweiligen System verfügbaren Laufwerke.

Das Verzeichnisfeld sollte unter anderem auch die Datei *qkarten.bas* enthalten. Öffnen Sie diese durch Eingabe ihres Namens in das Textfeld, oder wählen Sie diese mit Hilfe des Verzeichnisfeldes wie folgt aus:

1. Drücken Sie die TAB-TASTE, um den Cursor in das Verzeichnisfeld **Dateien** zu bringen.
2. Betätigen Sie die RICHTUNGSTASTEN, bis die gewünschte Programmbezeichnung ausgewählt wird. (Drücken Sie einfach die LEERTASTE, wenn das gewünschte Programm den ersten Eintrag bildet.)
3. Wählen Sie die Bezeichnung *qkarten.bas*, die **.bas* im Textfeld ersetzt.
4. Laden Sie *qkarten.bas* ins Arbeitsfenster durch Betätigen der EINGABETASTE.

Da *qkarten.bas* ziemlich umfangreich ist (fast 40K), ist die Ladezeit etwas länger. Nach Abschluß des Ladevorgangs, erscheint die Bezeichnung *qkarten.bas* in der Titelleiste des Arbeitsfensters.

Bewegen des Cursors

Der Cursor läßt sich in einem Fenster mit Hilfe der zum numerischen Tastenblock gehörenden RICHTUNGSTASTEN bewegen (die Pfeiltasten NACH LINKS(←), NACH RECHTS(→), NACH OBEN(↑) und NACH UNTEN(↓) sowie die Tasten POS1, ENDE, BILD↑ und BILD↓). Die RICHTUNGSTASTEN bewegen den Cursor um je eine Spalte oder Zeile. Die Tasten POS1 und ENDE bringen den Cursor zu der ersten oder letzten Spalte der aktuellen Zeile. Die STRG-TASTE kann jedoch mit den RICHTUNGSTASTEN kombiniert werden, um den Cursor in größeren Schritten wie folgt zu bewegen:

Tastenkombination	*Cursorbewegung*
STRG+NACH RECHTS	Ein Wort nach rechts
STRG+NACH LINKS	Ein Wort nach links
STRG+POS1	Erste Zeile des aktuellen Fensters
STRG+ENDE	Letzte Zeile des aktuellen Fensters

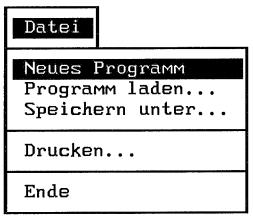
Rollen des Textes

Ein Fenster kann weitaus mehr Text enthalten, als innerhalb der Bildschirmgrenzen zu sehen ist. "Rollen" beschreibt die Textbewegung in und aus dem begrenzten Bildschirmbereich. Der Text läßt sich mit Hilfe nachstehender Tasten und Tastenkombinationen rollen:

Tastenkombination	*Rollvorgang*
BILD↑	Um eine gesamte Bildschirmanzeige nach oben
BILD↓	Um eine gesamte Bildschirmanzeige nach unten
STRG+BILD↓	Um eine gesamte Bildschirmanzeige nach rechts
STRG+BILD↑	Um eine gesamte Bildschirmanzeige nach links

Hinweis Die Tastenkombination STRG+BILD↓ kann verwirrend sein, wenn kein Text zwischen den Spalten 79 bis 159 vorhanden ist. Zur Bestätigung der jeweiligen Position in der Datei sind die Zeilen- und Spaltenzähler in der unteren rechten Ecke der Referenzzeile zu überprüfen. Der Cursor kann mit Hilfe der POS1-TASTE in die erste Spalte der aktuellen Zeile gebracht werden.

Starten eines neuen Programms



Beim Laden eines bestehenden Programms von Diskette löscht QuickBASIC den Arbeitsspeicher vor dem Ladevorgang. In diesem Abschnitt werden Sie *qkarten.bas* aus dem QuickBASIC-Arbeitsspeicher mit Hilfe des Befehls **Neues Programm** löschen und anschließend ein einfaches Programm eingeben. Es ist während der Eingabe von Code zu beachten, daß QuickBASIC die Schlüsselwörter der Anweisungen bei Betätigen der EINGABETASTE großschreibt. Wenn eine Fehlermeldung erscheint, löschen Sie diese mit der ESC-TASTE, und wiederholen Sie die Eingabe der Zeile an der Cursorstelle.

Löschen Sie *qkarten.bas* mit Hilfe nachstehender Arbeitsschritte. Schreiben und führen Sie dann einige Codezeilen aus:

1. Wählen Sie den Befehl **Neues Programm** aus dem Menü **Datei** durch Betätigen der Kombination ALT+D und der Taste N.

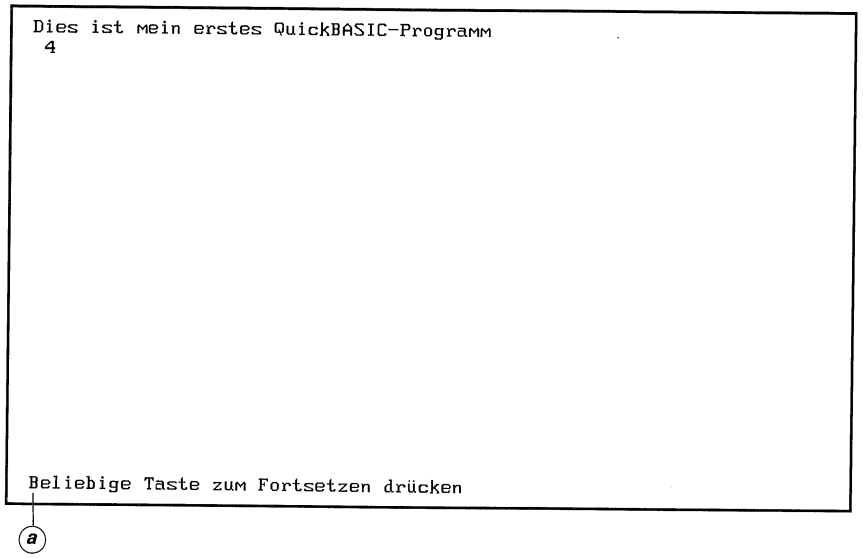
2. Geben Sie folgende BASIC-Anweisungen ein:

```
cls
PRINT "Dieses ist mein erstes QuickBASIC-Programm"
PRINT 2 + 2
```

3. Drücken Sie ALT+A und S, um den Befehl **Start** aus dem Menü **Ausführen** auszuwählen.

   Bei korrekter Eingabe zeigt QuickBASIC die Programmausgabe auf dem "Ausgabebildschirm" an, der wie in Abbildung 3.3 aussehen sollte.

Abbildung 3.3 Ausgabebildschirm für Codezeilen



a) Drücken Sie eine Taste, um in die QuickBASIC-Umgebung zurückzukehren.

In diesem Programm wird die Anzeige durch **CLS** gelöscht, und die **PRINT**-Anweisung zeigt den in Anführungsstrichen stehenden Text in der oberen linken Ecke des Bildschirms an. Da **PRINT** auch die Ergebnisse numerischer Ausdrücke anzeigen kann, erscheint die Summe des Ausdrucks `2+2` in der nächsten Zeile.

4. Kehren Sie in das QuickBASIC-Arbeitsfenster durch Betätigen einer beliebigen Taste zurück.
5. Drücken Sie ALT+N und A, um den Befehl **Ausgabebildschirm** aus dem Menü **Ansicht** zu wählen und sich das Programmergebnis erneut anzusehen.
6. Kehren Sie in das QuickBASIC-Arbeitsfenster durch Betätigen einer beliebigen Taste zurück.

Hinweis Hilfe für unverständliche Schlüsselwörter kann durch Positionieren des Cursors auf das Schlüsselwort und Drücken der F1-TASTE angefordert werden.

Ändern der Fenstergröße

Es ist einfach, die Fenster auf dem Bildschirm zu vergrößern oder zu verkleinern. Das aktive Fenster kann durch einmaliges Betätigen von ALT+PLUS um je eine Zeile vergrößert und durch einmaliges Drücken von ALT+MINUS um je eine Zeile verkleinert werden. (Es sind die PLUS und MINUS-TASTEN des numerischen Tastenblocks zu verwenden.)

Drücken Sie STRG+F10, um das aktive Fenster zeitweilig auf Gesamtbildschirmgröße zu erweitern. Die ursprüngliche Fenstergröße kann durch erneutes Betätigen dieses Umschalters wiederhergestellt werden. Nachstehende Arbeitsschritte veranschaulichen dieses Verfahren und bereiten die Fenster für die Übung im nächsten Abschnitt vor:

1. Drücken Sie ALT+PLUS zweimal, wobei sich der Cursor im Arbeitsfenster befindet.

   Dieses Verfahren fügt dem Arbeitsfenster Zeilen hinzu und komprimiert das darunterstehende Direktfenster.
2. Betätigen Sie ALT+MINUS siebenmal, um das Arbeitsfenster zu verkleinern.

   Dieses Verfahren reduziert die Zeilenanzahl des Arbeitsfensters und vergrößert gleichzeitig das Direktfenster.
3. Drücken Sie STRG+F10, um das aktuelle Fenster zeitweilig auf den Gesamtbildschirm zu erweitern.
4. Betätigen Sie erneut STRG+F10, um die ursprünglichen Proportionen der Anzeige wiederherzustellen.

Andere QuickBASIC-Fenster

Beim Starten von QuickBASIC und Löschen des einleitenden Dialogfeldes enthält das Arbeitsfenster den Cursor und ist somit das aktive Fenster. Es können nicht mehrere Fenster gleichzeitig aktiv sein, obwohl auch andere Fenster verfügbar sind. Es ist zu beachten, daß die hervorgehobene Titelleiste das aktive Fenster angibt.

Bewegen des Cursors zwischen den Fenstern

Der Cursor wird durch Betätigen der F6-TASTE in das darunterstehende Fenster und über die Tastenkombination UMSCHALT+F6 in das darüberstehende Fenster gebracht. Auch wenn das aktuelle Fenster bereits über die Tastenkombination STRG+F10 erweitert wurde, lassen sich das nächste und das vorhergehende Fenster weiterhin durch F6 bzw. UMSCHALT+F6 aktivieren.

Codeausführung im Direktfenster

Das Direktfenster am unteren Ende der Anzeige kann während der Programmiertätigkeit im Arbeitsfenster als Dienstfenster verwendet werden. Wenn Sie beispielsweise den Ausgabebildschirm löschen möchten, können Sie die **CLS**-Anweisung daraus ausführen. Das Direktfenster kann auch zum Durchführen von Berechnungen, Testen von Programmierideen und deren Ergebnissen benutzt werden. Beim Verlassen von QuickBASIC geht der im Direktenfenster vorhandene Text verloren.

Im Arbeitsfenster werden die Codezeilen sequentiell ausgeführt. Im Direktfenster wird ein einzeiliger Code ausgeführt, wenn der Cursor auf die betreffende Zeile positioniert und die EINGABETASTE betätigt wird. Im Direktfenster ist sowohl Groß- als auch Kleinschreibung zulässig. Beachten Sie jedoch, daß QuickBASIC im Direktfenster die Schlüsselworte nicht wie im Arbeitsfenster automatisch großschreibt.

Um die Codezeilen im Direktfenster auszuführen, verfahren Sie wie folgt:

1. Bringen Sie den Cursor in das Direktfenster durch Betätigen der F6-TASTE.
2. Geben Sie folgende Zeile ein:

   ```
   cls : PRINT "Diese Zeile kommt aus dem Direktfenster"
   ```

   Dieser Code besteht aus zwei, durch einen Doppelpunkt getrennten BASIC-Anweisungen. Der Doppelpunkt zwischen den Anweisungen ermöglicht den Einsatz mehrerer Anweisungen in einer Zeile.
3. Drücken Sie die EINGABETASTE, um den Code in dieser Zeile auszuführen.

   QuickBASIC zeigt den Ausgabebildschirm an.
4. Kehren Sie in die QuickBASIC-Umgebung über eine beliebige Taste zurück. Der Cursor wird auf der nächsten Leerzeile des Direktfensters plaziert.
5. Schreiben Sie die folgende Zeile genau wie angegeben:

   ```
   AktuelleZeit$ = ""
   ```

   Diese Zeile definiert eine als `AktuelleZeit$` bezeichnete Variable. Das `$`-Zeichen ist ein Hinweis für QuickBASIC, daß es sich um eine aus einer Zeichenfolge bestehende "Zeichenkettenvariable" handelt. Da es keinen Abstand zwischen den Anführungsstrichen gibt, hat die Zeichenkette die Länge Null.
6. Durch Betätigen der EINGABETASTE führen Sie diese Anweisung aus und weisen die Zeichenkette mit Länge Null der Variablen `AktuelleZeit$` zu.

Sie werden als Nächstes die Arbeit im Direktfenster fortsetzen und der Variablen `AktuelleZeit$` eine Zeichenkette mit der aktuellen Zeitangabe anhand der Funktion **TIME$** zuweisen.

1. Schreiben Sie folgende Zeile genau wie angegeben in die nächste Zeile des Direktfensters:

   ```
   AktuelleZeit$ = time$
   ```

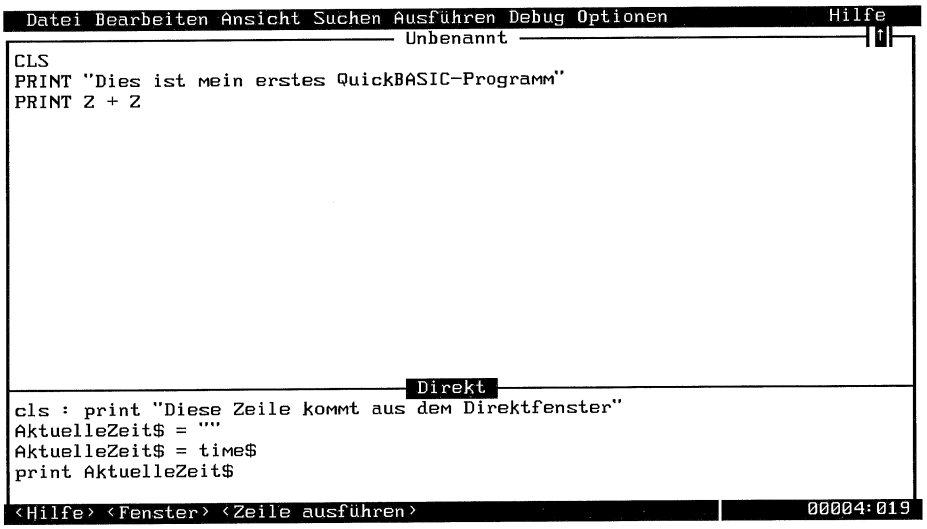
2. Führen Sie die eingegebene Zuweisungsanweisung durch Betätigen der EINGABETASTE aus.

3. Geben Sie folgende Zeile genau wie angegeben im Direktfenster ein:

   ```
   print AktuelleZeit$
   ```

   Abbildung 3.4 zeigt, wie das Direktfenster zu diesem Zeitpunkt aussehen sollte.

Abbildung 3.4 Das Direktfenster zeigt die eingegebenen Zeilen an



4. Drücken Sie die EINGABETASTE, um den Wert von `AktuelleZeit$` unter der ersten Zeile des Ausgabebildschirms anzuzeigen.

5. Kehren Sie in die Umgebung über eine beliebige Taste zurück.

   Im Direktfenster können bis zu zehn Anweisungszeilen geschrieben werden. Bei der Eingabe der 11. Zeile geht die 1. Zeile verloren, bei der Eingabe der 12. die 2. Zeile u.s.w. Abschnitt 10.5 veranschaulicht weitere Einsatzmöglichkeiten des Direktfensters.

Überwachen der Variablen mit Hilfe des Debug-Fensters

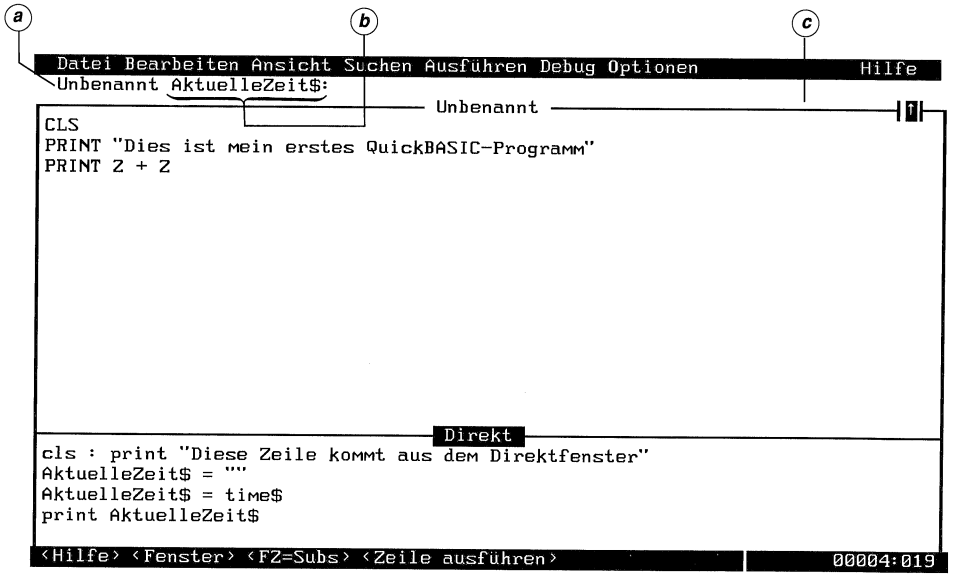
Das unerwartete Verhalten eines Programms liegt oft an einem unvorhergesehenen, von einer Variablen angenommenen Wert. Das Debug-Fenster ermöglicht das Überwachen der Ausdruckswerte während des Programmablaufes, um diese innerhalb des erwarteten Bereichs zu halten. Die Arbeitsweise des Debug-Fensters läßt sich wie folgt untersuchen:

1. Drücken Sie F6, um den Cursor, falls erforderlich, in das Direktfenster zu bringen.
2. Betätigen Sie ALT+G und Z, um den Befehl **Variable anzeigen** aus dem Menü **Debug** auszuwählen.

   Anschließend zeigt QuickBASIC das Dialogfeld **Variable anzeigen** an.
3. Geben Sie `AktuelleZeit$` ein, und betätigen Sie die EINGABETASTE.

   Das Debug-Fenster wird am oberen Ende des Bildschirms geöffnet. Es enthält die Bezeichnung aus der Titelleiste des Arbeitsfensters, die vom Namen der eingegebenen Variablen, `AktuelleZeit$`, gefolgt wird (siehe Abbildung 3.5). Der Cursor kehrt zum Direktfenster zurück.

Abbildung 3.5 Einsetzen von Variablen im Debug-Fenster



a) Bezeichnung aus der Titelleiste des Arbeitsfensters.

b) Variablenname.

c) Debug-Fenster.

4. Positionieren Sie den Cursor durch Betätigen der Tastenkombination STRG+POS1 auf die erste Zeile des Direktfensters und führen Sie dessen sämtliche Zeilen einzeln mit der EINGABETASTE aus.

   Betrachten Sie das Debug-Fenster, während Sie die EINGABETASTE auf jeder Zeile des Direktfensters betätigen. Sobald die Zeile `AktuelleZeit$ = Zeit$` ausgeführt wird, erscheint der Wert für `AktuelleZeit$` im Debug-Fenster.

Das Debug-Fenster ermöglicht das Überwachen von Variablenwerten während sich diese im Laufe einer Programmausführung ändern, ohne die Werte auf dem Ausgabebildschirm anhand von **PRINT**-Anweisungen anzeigen zu müssen.

Das Hilfsfenster

Die On-Line-Hilfe erscheint größtenteils im Hilfsfenster. Es gibt mehrere Möglichkeiten Hilfe für BASIC-Schlüsselworte, wie **PRINT** und **TIME$**, und für die QuickBASIC-Umgebung anzufordern. Sie können das Hilfsmenü verwenden oder F1 betätigen, während ein Bildschirmelement (oder ein Menü oder Befehl) den Cursor enthält oder hervorgehoben wird.

Hinweis Falls beim Zugriff auf die On-Line-Hilfe ein Fehler auftritt, sind nähere Informationen über die korrekte Installation der Hilfseinrichtung Abschnitt 10.8.4, "Die Hilfsdateien", zu entnehmen.

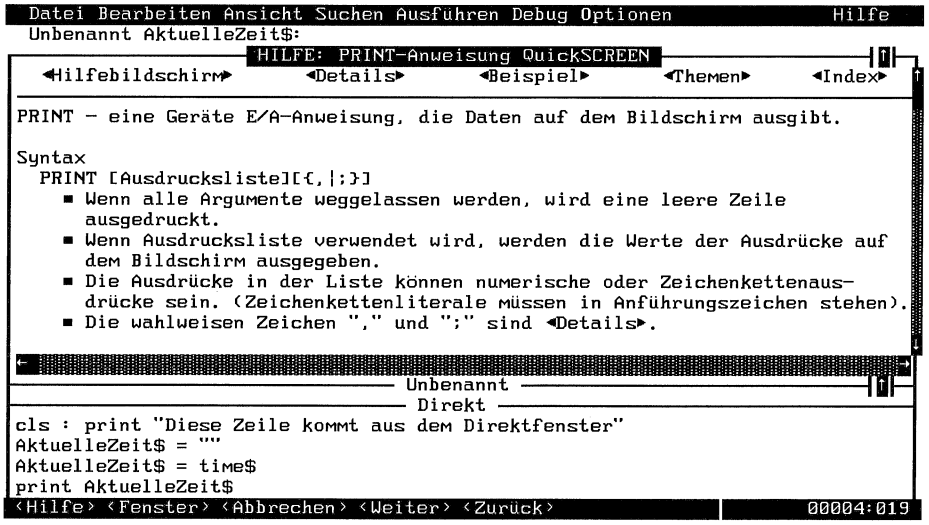
Kontextabhängige Hilfe

Anhand der kontextabhängigen Hilfe können Sie Informationen über BASIC-Schlüsselworte und Programmsymbole erhalten. Nachstehende Arbeitsschritte erläutern die Verwendung der kontextabhängigen Hilfe:

1. Schließen Sie das Hilfsmenü durch Betätigen der ESC-TASTE, falls erforderlich.
2. Bringen Sie den Cursor mit Hilfe der RICHTUNGSTASTEN zum Wort `PRINT`.
3. Drücken Sie F1.

   QuickBASIC öffnet einen Hilfebildschirm für das Schlüsselwort **PRINT** (siehe Abbildung 3.6).

Abbildung 3.6 Hilfebildschirm der Anweisung PRINT



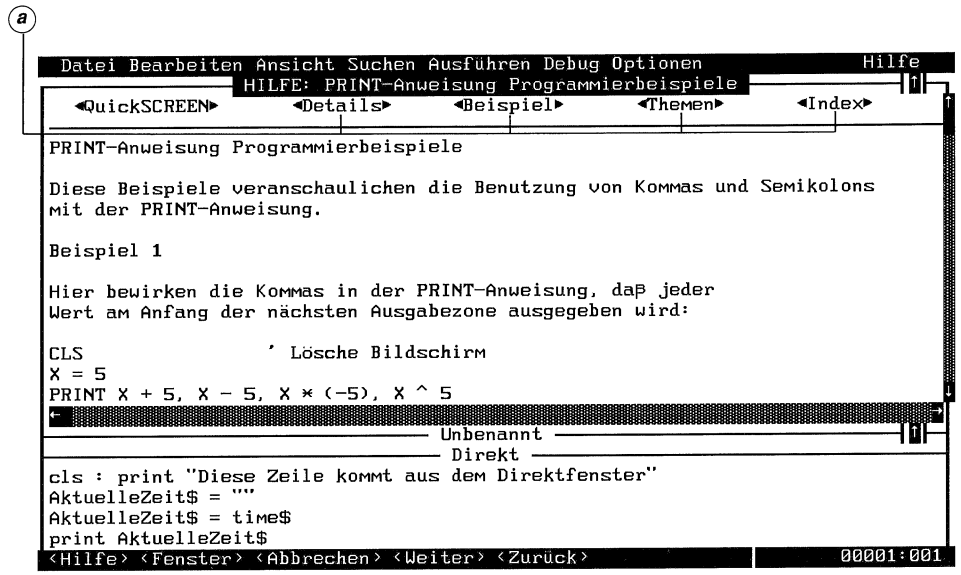
Der Hilfebildschirm liefert eine Kurzbeschreibung der Anweisungsfunktion, die deren Syntax angibt und deren Bestandteile erklärt. Am unteren Ende gibt der Hilfebildschirm weitere Anwendungsmöglichkeiten dieses Schlüsselwortes an.

Hyperverbindungen

Am oberen Ende des Bildschirms geben mehrere, als "Hyperverbindungen" bezeichnete, in Klammern eingeschlossene Elemente zusammenhängende Themen an, die ebenfalls untersucht werden können. Die Hyperverbindungen gewährleisten eine sofortige Verbindung zwischen zusammenhängenden Themen. Am unteren Ende des Bildschirms werden die anderen Anwendungen des Schlüsselwortes ebenfalls als Hyperverbindungen in Klammern eingeschlossen. Wenn Sie weitere Informationen über einen dieser Begriffe benötigen, bewegen Sie den Cursor anhand der TAB-TASTE zu der Hyperverbindung, und drücken Sie F1, wie nachstehend beschrieben:

1. Drücken Sie F6, bis der Cursor im Hilfsfenster erscheint.
2. Bringen Sie den Cursor durch Betätigen der TAB-TASTE zu der Hyperverbindung **Beispiel**.
3. Bringen Sie die Anzeige **Beispiel** mit Hilfe der Taste F1 auf den Bildschirm (siehe Abbildung 3.7).

Abbildung 3.7 Beispielanzeige für die Anweisung PRINT



a) Hyperverbindungen mit zusammenhängenden Themen.

Sie können sich im Hilfsfenster bewegen und alle **PRINT**-Beispiele ansehen.

4. Drücken Sie ALT+F1 ein- oder zweimal, um zu der ursprünglichen Anzeige zurückzukehren.
5. Wiederholen Sie die Arbeitsschritte 1 und 2, um die durch die anderen eingeklammerten Hyperverbindungen verfügbare Information durchzugehen.

Es können mehrere aufeinanderfolgende Hyperverbindungen verwendet werden, ohne zu der vorhergehenden Anzeige zurückzukehren, da QuickBASIC die letzten zwanzig Hyperverbindungen erfaßt. Anhand der Tastenkombination ALT+F1 können Sie die Reihenfolge der Hyperverbindungen zurückverfolgen. Wenn die ursprüngliche Anzeige erreicht wird, gibt QuickBASIC einen Piepton aus, sobald ALT+F1 betätigt wird. Das Hilfsfenster kann jederzeit durch Betätigen der ESC-TASTE vollständig gelöscht werden, auch wenn Sie bereits mehrere Ebenen der Hyperverbindungsanzeigen durchlaufen haben.

Als Gedächtnisstütze für die zum Durchgehen der Hilfsanzeigen erforderlichen Tasten und Tastenkombinationen können Sie **Themen** aus dem Menü **Hilfe** und anschließend die Hyperverbindung "Hilfetasten" auswählen. Eine allgemeine Erklärung der On-Line-Hilfe können Sie auch über die Tastenkombination UMSCHALT+F1 anfordern.

Einige nicht eingeklammerte Elemente stellen ebenfalls Hyperverbindungen dar. Drücken Sie F1 für alle Fragen bezüglich des Hilfebildschirms. Wenn QuickBASIC über weitere Informationen diesbezüglich verfügt, erscheinen diese im Hilfsfenster, wenn das nicht der Fall ist, gibt QuickBASIC einen Piepton aus.

Hinweis Im Hilfsfenster kann kein Text bearbeitet werden, jedoch kann Text sowohl im Direkt- als auch im Arbeitsfenster bearbeitet werden, während das Hilfsfenster geöffnet bleibt. Kapitel 8, "Verwendung von Beispielcode aus der On-Line-Hilfe", zeigt, wie Text aus dem Hilfsfenster kopiert und in das Programm eingefügt wird.

Verlassen der QuickBASIC-Umgebung und Speichern der Programme

Folgende Arbeitsschritte erklären, wie Sie QuickBASIC verlassen und in die DOS-Befehlszeile zurückkehren können:

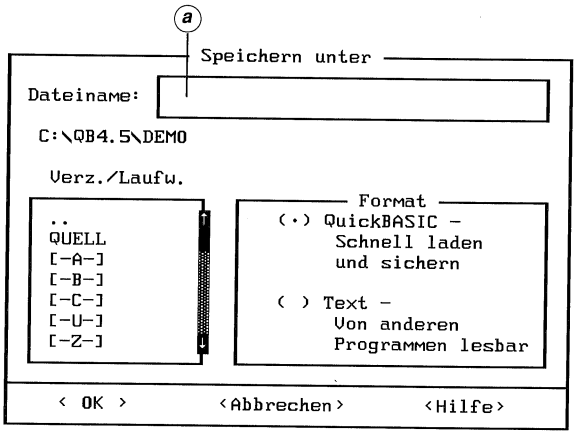
1. Drücken Sie ALT+D und E, um QuickBASIC zu verlassen.

   Ein Dialogfeld erscheint mit der Anfrage, ob im Arbeitsfenster das gesamte Material bereits gespeichert wurde.

2. Betätigen Sie die EINGABETASTE, um QuickBASIC mitzuteilen, daß das im Arbeitsfenster vorhandene Programm gespeichert werden soll.

   QuickBASIC zeigt das Dialogfeld **Speichern unter** an (siehe Abbildung 3.8):

Abbildung 3.8 Das Dialogfeld Speichern unter



a) Geben Sie den Namen des Programms an dieser Stelle ein.

3. Geben Sie im Textfeld `Programm1` ein.
4. Speichern Sie die im Arbeitsfenster vorhandenen Zeilen durch Betätigen der EINGABETASTE auf Diskette/Festplatte unter dem Namen *programm1.bas*.

Weitere Informationen

Weitere Informationen zu den in diesem Kapitel erläuterten Themen sind nachstehenden Abschnitten zu entnehmen:

Abschnitt	*Information*
Kapitel 9, "Fehlerbeseitigung während der Programmierung", und Kapitel 17, "Grundkonzepte und Techniken der Fehlerbeseitigung"	Techniken zur Fehlerbeseitigung und Verwendung des Debug-Fensters
Abschnitt 10.5, "Verwendung des Direktfensters"	Anwendungsmöglichkeiten des Direktfensters
Kapitel 11, "Das Menü Datei"	Öffnen, Löschen und Speichern der Dateien in QuickBASIC

4 BASIC für Anfänger

Dieses Kapitel führt die allgemeinen Programmiergrundsätze der BASIC-Sprache ein und konzentriert sich auf die grundlegenden Begriffe, um den Programmieranfänger in die Lage zu versetzen, so schnell wie möglich mit dem Programmieren zu beginnen.

Demnach ist es für jeden Programmieranfänger empfehlenswert, dieses Kapitel sorgfältig durchzulesen, um nach Abschluß einfache BASIC-Programme schreiben zu können. Es ist ebenfalls ratsam, vor dem Durcharbeiten der in den nächsten Kapiteln vorgestellten fortgeschrittenen Programmiertechniken, einige andere BASIC-Bücher für fortgeschrittene Programmierer zu lesen.

Wenn Sie bereits in BASIC programmiert haben, ist es ebenfalls empfehlenswert, dieses Kapitel durchzublättern, um sich einen Überblick über die Unterschiede zwischen Microsoft QuickBASIC und der Ihnen bekannten Abwandlung der BASIC-Sprache zu verschaffen.

Zahlreiche kurze Programmierbeispiele veranschaulichen die Arbeitsweise jedes Elements. Die Eingabe und Ausführung dieser Beispiele beim Durchlesen des Kapitels bilden die beste Methode, BASIC schnell zu erlernen. Im Arbeitsfenster können mehrzeilige Beispiele eingegeben werden, während im Direktfenster von QuickBASIC einzeilige Beispiele eingegeben und die entsprechende Ausgabe durch Betätigen der EINGABETASTE auf den Bildschirm gebracht werden können. Bearbeiten Sie jedes Beispiel, um dessen Funktionsweise zu verstehen, und eignen Sie sich das Material eines Abschnitts an, bevor Sie mit dem nächsten fortfahren.

Nach Abschluß dieses Kapitels werden Sie in der Lage sein, folgende Operationen durchzuführen:

- Darstellung von Wörtern und Zahlen auf dem Bildschirm.
- Definition von Variablen und deren Verwendung.
- Durchführung von Berechnungen und Anwendung der mathematischen BASIC-Funktionen.
- Verwendung von Datenfeldern.
- Einzelne Ausführung von Programmabschnitten.
- Wiederholung einer Gruppe von Anweisungen sooft wie angegeben oder bis eine Bedingung wahr wird.

Programmdefinition

Ein "Programm" besteht aus einer Folge von Anweisungen an den Computer in einer sowohl dem Computer als auch dem Benutzer verständlichen Sprache. Jede einzelne Anweisung beansprucht normalerweise eine Programmzeile.

Das Ziel jedes Programms ist die Durchführung einer nützlichen Tätigkeit, wie zum Beispiel Textverarbeitung, Buchführung oder ein Spiel. Programmieren bedeutet, zu entscheiden, was das Programm zu tun hat. Anschließend werden Sprachanweisungen ausgewählt und angeordnet, um die einzelnen, zum Erreichen des allgemeinen Ziels erforderlichen Aufgaben durchzuführen. Diese Tätigkeit kombiniert Fachkenntnis und Erfahrung. Die Fachkenntnisse werden aus Büchern erlernt, die Erfahrung wird durch Schreiben eigener Programme und Analysieren anderer Programme erworben.

Zahlreiche BASIC-Anweisungen weisen zusätzliche, in diesem Kapitel nicht beschriebene Eigenschaften auf. Weitere Informationen über eine bestimmte Anweisung oder Spracheigenschaft sind Kapitel 8, "Zusammenfassung der Anweisungen und Funktionen", des Handbuchs *Programmieren in BASIC* zu entnehmen oder von der On-Line-Hilfe anzufordern.

Kommentare

Kommentare bilden einen bedeutenden Teil der Programmiertätigkeit, da sie das Programm erklären und den Programmierer an die eigene Vorgehensweise bei der Erstellung erinnern. Kommentare beginnen mit einem Apostroph und können wie folgt bis zum Ende der Zeile gehen:

```
' Das ist ein Kommentar.
CLS          ' Die CLS-Anweisung löscht den Bildschirm.
```

Der Apostroph teilt QuickBASIC mit, daß der Text zwischen dem Apostroph und dem Ende der Zeile einen Kommentar bildet.

Darstellung von Wörtern und Zahlen auf dem Bildschirm

Wahrscheinlich ist Ihnen Anwendungssoftware, wie Lotus 1-2-3 oder Microsoft® Word bereits bekannt. Diese und die meisten anderen Programme stehen in Verbindung mit dem Benutzer durch die Darstellung von Wörtern, Zahlen und Daten auf dem Bildschirm.

Die **PRINT**-Anweisung bildet den BASIC-Befehl für die Ausgabe auf dem Bildschirm. **PRINT** kann sowohl Zeichenketten als auch Zahlen ausgeben.

Eine "Zeichenkette" besteht aus einer zwischen Anführungsstrichen eingeschlossenen Zeichenfolge (Buchstaben, Ziffern, Satzzeichen). Das folgende Programmbeispiel verdeutlicht die Anzeige einer Zeichenkette mit Hilfe der Anweisung **PRINT**:

1. Starten Sie QuickBASIC, falls erforderlich, indem Sie `qb` eingeben und die EINGABETASTE betätigen.
2. Tippen Sie folgende Zeile im Direktfenster ein und drücken Sie die EINGABETASTE:

   ```
   PRINT "Ich grüße den Rest der Welt"
   ```

   Die Ausgabe sollte wie folgt aussehen:

   ```
   Ich grüße den Rest der Welt
   ```

   Die Anführungsstriche, die eine Zeichenkette einschließen, gehören nicht dazu und werden somit nicht ausgegeben.
3. Geben Sie `CLS` im Direktfenster ein, und löschen Sie den Ausgabebildschirm durch Betätigen der EINGABETASTE.

Hinweis Da BASIC auf Groß- und Kleinschreibung nicht achtet, braucht die Schreibweise der Beispiele nicht eingehalten werden. QuickBASIC schreibt alle zur BASIC-Sprache gehörenden Wörter automatisch groß und beläßt den restlichen Text wie eingegeben. Die Beispiele dieses Kapitels werden wie von QuickBASIC formatiert und Programmausgaben so, wie sie auf dem Bildschirm erscheinen, dargestellt.

Die Anweisung **PRINT** kann, wie im nachstehenden Beispiel, auch Zahlen ausgeben:

1. Geben Sie folgende Zeile im Direktfenster ein, und betätigen Sie die EINGABETASTE:

   ```
   PRINT 63
   ```

   Anschließend erscheint folgende Ausgabe:

   ```
    63
   ```
2. Löschen Sie den Ausgabebildschirm, indem Sie `CLS` eingeben und die EINGABETASTE betätigen.

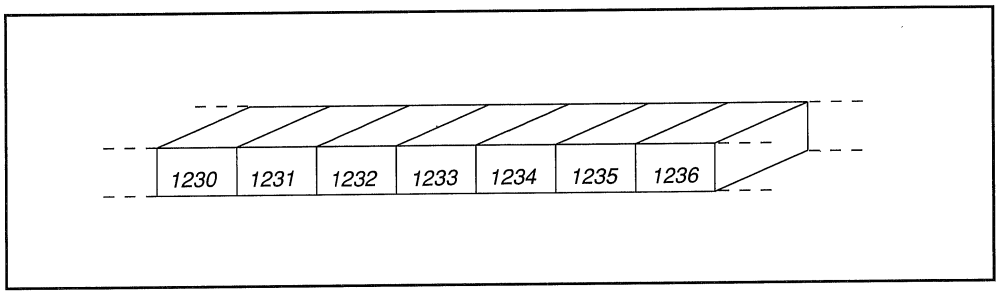
Variablen

Programme erhalten Daten von der Außenwelt (Eingabe) und verarbeiten diese zu einer nützlichen Information (Ausgabe). Sowohl Ein- als auch Ausgabedaten werden im Speicher des Computers abgelegt. Der Speicherplatz für Daten wird mit Hilfe von Variablenbezeichnungen belegt.

Speichern von Daten

Der Speicher besteht aus Bereichen, die eine jeweils eindeutige "Adresse" aufweisen. Die Adresse besteht ihrerseits aus einer Zahl, die den Speicherplatz lokalisiert, genau wie eine Hausadresse die Lage des Hauses in einer Straße angibt. Jede einzelne Information hat ihren eigenen Speicherplatz und eine entsprechende Adresse (siehe Abbildung 4.1).

Abbildung 4.1 Adressen in einem Speicher



Zum Glück brauchen Sie nicht die Adressen aller Datenelemente zu erfassen. BASIC verwendet einen "Variablennamen", um das gewünschte Wort mit dem Speicherplatz im Computer zu verbinden.

BASIC stellt diese Verbindung automatisch her. Beim Auftreten eines unbekannten Variablennamens sucht BASIC einen ungenutzten Speicherplatz und verbindet den neuen Variablennamen mit dieser Adresse. Auf diese Weise kann der Variablenname benutzt werden, ohne dessen Adresse zu kennen.

Das folgende Beispielprogramm demonstriert diesen Vorgang. Es ist zu beachten, daß `Gruess Dich, Welt` in diesem Beispiel nicht in Anführungsstrichen steht.

1. Geben Sie folgende Zeile im Direktfenster ein, und betätigen Sie die EINGABETASTE:

   ```
   PRINT Gruess Dich, Welt
   ```

   Es erscheint folgende Ausgabe:

   ```
    0  0            0
   ```

2. Tippen Sie `CLS` im Direktfenster ein, und löschen Sie den Ausgabebildschirm durch Betätigen der EINGABETASTE:

BASIC muß die Worte `Gruess`, `Dich` und `Welt` definieren. Es handelt sich nicht um Schlüsselworte wie **PRINT** (das bedeutet, daß sie nicht zu der BASIC-Sprache gehören) oder um Zeichenketten, da sie nicht mit Anführungsstrichen versehen sind, und ganz bestimmt nicht um Zahlen. Demnach müssen `Gruess`, `Dich` und `Welt` Variablen sein.

Nach Fällen dieser Entscheidung belegt BASIC Speicherplatz für drei neue Variablen mit den Namen `Gruess`, `Dich` und `Welt`. Da neuen Variablen stets ein Nullanfangswert zugeordnet wird, zeigt die Anweisung **PRINT** drei Nullen an.

Ein Variablenname kann Ziffern enthalten, aber er darf nicht mit einer Ziffer beginnen, da jedes mit einer Ziffer beginnende Wort als Zahl betrachtet wird. Beachten Sie, daß in Variablennamen die deutschen Zeichen Ä, ä, Ö, ö, Ü, ü und ß nicht zulässig sind.

Die in BASIC zugelassenen Namen können wie folgt aussehen:

```
abcdef
GeburtsDatum
Name3
ep451p33
```

sind aber in dieser Form unzulässig:

```
5entfernen
4.3weniger
ändern
Grüße
```

Variablentypen

Daten können mehrere Formen annehmen. Aus diesem Grund benötigen Programmiersprachen unterschiedliche Variablentypen, um verschiedene Datentypen zu erfassen. BASIC weist drei Variablenkategorien auf: Ganzzahlen, Gleitkommazahlen und Zeichenketten.

Ganzzahlige Variablen

Eine Ganzzahl besteht aus einer ganzen Zahl ohne Bruchteile. 0, -45 und 1278 sind zum Beispiel Ganzzahlen. Ganzzahlen werden in ganzzahligen Variablen gespeichert, die ihrerseits von BASIC in zwei Kategorien eingeteilt werden: Ganzzahlen und lange Ganzzahlen.

Ein Variablenname, der mit dem Zeichen **%** endet, bildet eine ganzzahlige Variable, die jeden Wert zwischen -32 768 und 32 767 annehmen kann. Ein Variablenname, der mit dem Zeichen **&** endet, bildet eine lange ganzzahlige Variable und kann jeden Wert zwischen -2 147 483 648 und 2 147 483 647 annehmen.

Gleitkommavariablen

Eine Gleitkommazahl kann Stellen hinter dem Komma aufweisen. 0.123, 78.26, 7.0 und 1922.001234 sind beispielsweise Gleitkommazahlen. Gleitkommazahlen werden in Gleitkommavariablen gespeichert. BASIC teilt die Gleitkommavariablen in zwei Kategorien ein: Gleitkommavariablen einfacher Genauigkeit und Gleitkommavariablen doppelter Genauigkeit.

Ein Variablenname, der mit einem Buchstaben, einer Ziffer oder dem Zeichen **!** endet, bildet eine Gleitkommavariable einfacher Genauigkeit. Gleichkommazahlen einfacher Genauigkeit können Werte zwischen -3,4 x 10^{38} und 3,4 x 10^{38} (34 gefolgt von 37 Nullen) annehmen. Eine Gleitkommazahl einfacher Genauigkeit ist auf nur 7 Dezimalstellen genau.

Ein Variablenname, der mit dem Zeichen # endet, bildet eine Gleitkommavariable doppelter Genauigkeit und kann Werte zwischen -1,7 x 10^{308} und 1,7 x 10^{308} (17 gefolgt von 307 Nullen!) annehmen. Eine Gleitkommavariable doppelter Genauigkeit kann somit viel größere Zahlen umfassen als eine Variable einfacher Genauigkeit und ist auf etwa 15 Dezimalstellen genau.

In Berechnungen werden am häufigsten Gleitkommazahlen einfacher Genauigkeit verwendet, da "wirklichkeitsnahe" Anwendungen normalerweise Zahlen mit Dezimalstellen (zum Beispiel für Preise) erfordern. Aus diesem Grund bilden die Gleitkommavariablen einfacher Genauigkeit den "Standard"-Datentyp. Mit anderen Worten, wenn ein Variablenname nicht mit einem "Typenzeichen" (**%**, **&**, **^**, **!** oder **$**) endet, wird vorausgesetzt, daß es sich um eine Gleitkommavariable einfacher Genauigkeit handelt.

Es ist jedoch zu beachten, daß Berechnungen mit ganzzahligen Variablen erheblich schneller durchgeführt werden als die mit Gleitkommavariablen. Wenn demnach die Geschwindigkeit von Bedeutung ist, sollten ganzzahlige Variablen verwendet werden.

Die numerischen Variablen in den restlichen Beispielen dieses Kapitels werden mit den entsprechenden Typensuffixen **!** oder **%** gekennzeichnet. Obwohl es sich in den meisten Fällen um Gleitkommavariablen einfacher Genauigkeit handelt, die keinen Datensuffix erfordern, wird der Suffix **!** zur Verdeutlichung hinzugefügt.

Zeichenkettenvariablen

Eine Zeichenkettenvariable enthält eine Folge (Kette) von Zeichen. Ein Variablenname, der mit einem **$**-Zeichen endet, bildet eine Zeichenkettenvariable. Genau wie einer numerischen Variablen ein Nullanfangswert zugeordnet ist, wird auch einer neu definierten Zeichenkettenvariablen eine Nullanfangslänge zugewiesen. Sie enthält ursprünglich überhaupt keine Daten, nicht einmal Leerstellen. Wenn der Variablen `Leerstelle$` kein Wert zugewiesen wurde, zeigt die Anweisung

```
PRINT Leerstelle$
```

eine Leerzeile an.

Zuweisung von Werten an Variablen

Einer Variablen wird ein Wert anhand einer "Zuweisungsanweisung" zugeordnet. Eine Zuweisungsanweisung besteht aus drei Elementen: der Variablen, die einen neuen Wert enthält, einem Gleichheitszeichen (=) und der Zahl, Zeichenkette oder Berechnung, deren Wert die Variable annimmt. Nachstehend werden einige gültige Zuweisungsanweisungen dargestellt:

```
Alter% = 41
MeinName$ = "Willi"
Ergebnis! = Ergebnis3! + LiWert! + ReWert!
```

Im nächsten Beispielprogramm werden zwei Zeichenketten mit einem Pluszeichen (+) kombiniert, um eine längere Zeichenkette zu erhalten. Das Ergebnis wird einer dritten Zeichenkette zugewiesen.

1. Wählen Sie den Befehl **Neues Programm** aus dem Menü **Datei**, um das Arbeitsfenster zu löschen (drücken Sie ALT+D und N).
2. Tippen Sie folgendes Beispiel im Arbeitsfenster ein:

   ```
   ZeichKett1$ = "Guten"
   ZeichKett2$ = " Tag!"
   LangeZeichKett$ = Zeichkett1$ + Zeichkett2$
   PRINT LangeZeichkett$
   ```

3. Wählen Sie den Befehl **Start** aus dem Menü **Ausführen** (drücken Sie ALT+A und S).

   Die Ausgabe muß wie folgt aussehen:

   ```
   Guten Tag!
   ```

4. Geben Sie `CLS` im Direktfenster ein, und löschen Sie den Ausgabebildschirm durch Betätigen der EINGABETASTE.

Obwohl die Zuweisung durch das Gleichheitszeichen angegeben wird, stellt sie keine algebraische Gleichheit dar. Die Zuweisungsanweisung teilt BASIC folgendes mit: "Führe alle Berechnungen und Datenmanipulationen auf der rechten Seite des Gleichheitszeichens durch, und weise anschließend das Ergebnis der Variablen auf der linken Seite zu".

Der gleiche Variablenname kann auf beiden Seiten der Zuweisungsanweisung erscheinen. Wie aus nachstehendem Beispielprogramm ersichtlich, tritt dieser Vorgang meistens auf, wenn der Variablenwert geändert werden soll.

1. Wählen Sie den Befehl **Neues Programm** aus dem Menü **Datei** aus, um das Arbeitsfenster zu löschen.
2. Geben Sie folgendes Beispiel im Arbeitsfenster ein:

```
Zähler% = 6
PRINT Zähler%
Zähler% = Zähler% - 1
PRINT Zähler%
```

3. Wählen Sie den Befehl **Start** aus dem Menü **Ausführen**.

   Die Ausgabe sieht wie folgt aus:

```
6
5
```

4. Geben Sie `CLS` im Direktfenster ein, und löschen Sie die Bildschirmausgabe durch Betätigen der EINGABETASTE.

Zeichenkettenvariablen können keine numerische Werte zugewiesen werden (und umgekehrt). Demnach sind folgende Anweisungen unzulässig und unterbrechen die Ausführung oder führen zu der Fehlermeldung `Unverträgliche Datentypen` während der Programmausführung.

```
ZeichKe$ = 3.14159
Zahl! = "Guten Tag"
```

Berechnungen

In BASIC werden die vier mathematischen Operationen Addition, Subtraktion, Multiplikation und Division durch die Zeichen +, -, * (Stern) und / (Schrägstrich) dargestellt. Die Potenzierung (eine Zahl zu einer Potenz erheben) wird durch das Dachzeichen ^ wiedergegeben. Folgende Daten können auf je einer Zeile des Direktfensters eingegeben und die Ergebnisse der Operationen durch Betätigen der EINGABETASTE ausgegeben werden:

Operation	*Ergebnis*
`PRINT 2 + 3`	`5`
`PRINT 2 - 3`	`-1`
`PRINT 2 * 3`	`6`
`PRINT 2 / 3`	`0.6666667`
`PRINT 2 ^ 3`	`8`

Ganzzahlige Division und Restoperator

BASIC verfügt über zwei mathematische Operationen, ganzzahlige Division und Restoperator, die Ihnen wahrscheinlich unbekannt sind.

Bei der ganzzahligen Division wird der ganzzahlige Teil der Division behalten und der Bruchteil abgeschnitten. Die ganzzahlige Division wird durch das Zeichen \ (rückwärtiger Schrägstrich) dargestellt. Sie können folgende Daten auf je einer Zeile des Direktfensters eingeben und die EINGABETASTE betätigen, um die Ergebnisse der Operationen zu untersuchen:

Operation	*Ergebnis*
`PRINT 7 \ 3`	`2`
`PRINT 9.6 \ 2.4`	`5`

Sowohl die ganzzahlige Division als auch der Restoperator runden die betreffenden Zahlen vor Durchführung der Berechnung. Die Zahl `9.6` wird `10` und `2.4` wird `2`. Aus diesem Grund ist das Ergebnis des 2. obenstehenden Beispiels 5 anstatt 4.

Der Restoperator ist komplementär zur ganzzahligen Division. In diesem Fall wird der ganzzahlige Teil der Division entfernt und der Rest behalten. Die Restdivision wird mit Hilfe des BASIC-Schlüsselwortes **MOD** durchgeführt. Sie können die folgenden Daten auf je einer Zeile des Direktfensters eingeben und sich die Ergebnisse der Operationen mit Hilfe der EINGABETASTE ansehen:

Operation	*Ergebnis*
`PRINT 7 MOD 3`	`1`
`PRINT 11 MOD 4`	`3`

Vorrang der Operationen

BASIC wertet mathematische Ausdrücke von links nach rechts nach den Regeln des algebraischen Vorrangs aus: zuerst wird die Potenzierung durchgeführt, dann die Multiplikation und Division und zuletzt die Addition und Subtraktion. Das folgende Beispiel veranschaulicht den algebraischen Vorrang:

1. Geben Sie folgende Zeile im Direktfenster ein, und betätigen Sie die EINGABETASTE:

   ```
   PRINT 2 * 3 + 2 ^ 3 - 2 / 3
   ```

   Die Ausgabe erscheint wie folgt:

   ```
   13.33333
   ```

2. Geben Sie `CLS` im Direktfenster ein, und löschen Sie den Ausgabebildschirm durch Betätigen der EINGABETASTE.

Klammern können den normalen Vorrang bestätigen oder aufheben, wie in diesem Beispiel:

1. Tippen Sie folgende Zeile im Direktfenster ein, und drücken Sie die EINGABETASTE:

   ```
   PRINT (2 * 3) + (2 ^ 3) - (2 / 3)
   ```

   Die Ausgabe sollte wie folgt aussehen:

   ```
   13.33333
   ```

2. Geben Sie `CLS` im Direktfenster ein, und drücken Sie die EINGABETASTE, um den Ausgabebildschirm zu löschen.

Das Ergebnis ist dasselbe wie im vorhergehenden Beispiel, da der von den Klammern festgelegte Vorrang der Operationen nicht von der normalen Reihenfolge abweicht. Das nächste Beispielprogramm hat jedoch ein unterschiedliches Ergebnis zur Folge:

1. Geben Sie folgende Zeile im Direktfenster ein, und betätigen Sie die EINGABETASTE:

   ```
   PRINT 2 * (3 + 2) ^ (3 - 2 / 3)
   ```

   Die Ausgabe erscheint wie folgt:

   ```
   85.49879
   ```

2. Geben Sie `CLS` im Direktfenster ein, und löschen Sie den Ausgabebildschirm anhand der EINGABETASTE.

Das Ergebnis ist verschieden, da die eingeklammerten Berechnungen zuerst ausgeführt werden. Innerhalb der Klammern ist natürlich der Vorrang weiterhin gültig. Der Ausdruck `(3 - 2 / 3)` wird als `(3 - 0.6666667)`, nicht `(1 / 3)`, bewertet.

Um die Reihenfolge der Berechnungen zu steuern, empfiehlt es sich, Klammern zu verwenden, anstatt auf den Vorrang zu achten. Klammern vermeiden Fehler und machen Programme übersichtlicher.

Mathematische Funktionen

Zusätzlich zu den bereits vorgestellten Additions-, Subtraktions-, Multiplikations- und Divisionsoperatoren, bietet BASIC eine Reihe von häufig verwendeten mathematischen Funktionen, die nachstehend kurz beschrieben werden. Um diese anzuwenden, geben Sie die Variable oder den Ausdruck, mit der/dem die Funktion zu benutzen ist, nach dem Variablennamen ein. Schließen Sie die Variable oder den Ausdruck in Klammern ein. Der Ausdruck kann aus jeder gültigen Kombination von Variablen, Zahlen und mathematischen Funktionen bestehen. (Weitere Erläuterungen zu Ausdrücken sind im nächsten Abschnitt zu finden.)

Funktion	*Beschreibung*
ABS	Liefert den absoluten Wert des Ausdrucks
ATN	Liefert den Winkel, dessen Tangens gleich dem Ausdruck ist, im Bogenmaß
COS	Liefert den Kosinus eines Winkels im Bogenmaß
EXP	Liefert *e* potenziert mit Ausdruck
LOG	Liefert den Logarithmus des Ausdrucks
SGN	Liefert -1 zurück, wenn der Ausdruck kleiner als 0 ist, 0, wenn er gleich 0 ist und +1, wenn er größer als 0 ist
SIN	Liefert den Sinus eines Winkels im Bogenmaß
SQR	Liefert die Quadratwurzel des Ausdrucks
TAN	Liefert den Tangens eines Winkels im Bogenmaß

Das folgende Beispiel veranschaulicht die Anwendung der Quadratwurzelfunktion:

1. Löschen Sie das Arbeitsfenster mit Hilfe des Befehls **Neues Programm** aus dem Menü **Datei**.
2. Geben Sie folgendes Beispeil im Arbeitsfenster ein:

```
Benachbart! = 3.0
Entgegengesetzt! = 4.0
Hypotenuse! = SQR ((Benachbart! ^ 2) + _
   (Entgegengesetzt! ^ 2))
PRINT "die Hypotenuse ist:"; Hypotenuse!
```

3. Wählen Sie den Befehl **Start** aus dem Menü **Ausführen** aus.

   Die Ausgabe erscheint wie folgt:

```
Die Hypotenuse ist: 5
```

4. Geben Sie im Direktfenster `CLS` ein, und drücken Sie die EINGABETASTE, um den Ausgabebildschirm zu löschen.

Ausdrücke

Ein "Ausdruck" kann am besten mit einer Formel verglichen werden. Ein Ausdruck ist aus einer auswertbaren Kombination von Zahlen, Zeichenketten, Variablen, Funktionen und Operatoren zusammengesetzt. Zum Beispiel ist `2 + 2` ein einfacher Ausdruck, der als vier ausgewertet wird. Geben Sie die folgenden Anweisungen im Arbeitsfenster ein, um eine Reihe einfacher Ausdrücke Variablen wie folgt zuzuweisen:

1. Löschen Sie das Arbeitsfenster mit Hilfe des Befehls **Neues Programm** aus dem Menü **Datei**.
2. Geben Sie folgendes Beispiel im Arbeitsfenster ein:

```
Quadrat% = 3 ^ 2
Wurzel! = SQR(3 ^ 2)
Grüße$ = "Grüß Dich," + " Welt! "
Seltsam! = 9 + "Katzen"
```

   Die zuletzt eingegebene Anweisung ist unzulässig.
3. Wählen Sie den Befehl **Start** aus dem Menü **Ausführen** aus.

   QuickBASIC zeigt die Fehlermeldung `Unverträgliche Datentypen` an, da eine Zahl (oder numerische Variable) nicht zu einer Zeichenkette (oder Zeichenkettenvariablen) addiert werden kann.
4. Geben Sie `CLS` im Direktfenster ein, und löschen Sie den Ausgabebildschirm durch Betätigen der EINGABETASTE.

Ausdrücke können jede gewünschte Komplexität aufweisen, wobei die Ergebnisse eines Ausdrucks an eine Funktion übergeben oder der von einer Funktion zurückgegebene Wert mit anderen Ausdrücken kombiniert werden kann, usw.

An fast jeder Stelle, an der eine Zahl oder Zeichenkette benutzt wird, kann auch ein beliebig komplexer Ausdruck verwendet werden, der zu einer Zahl oder einer Zeichenkette ausgewertet wird. Das bedeutet, daß Ausdrücke in mathematischen BASIC-Funktionen, **PRINT**- und Zuweisungsanweisungen zur Anwendung kommen können.

Ausgabe von Variablen und Ausdrücken

Die **PRINT**-Anweisung wird nicht nur, wie am Anfang dieses Kapitels gezeigt, zur Ausgabe von Zahlen und Zeichenketten eingesetzt, sie kann auch Variablen- und Ausdruckswerte wie folgt ausgeben:

1. Geben Sie die folgenden Zeilen im Direktfenster ein, und drücken Sie die EINGABETASTE:

   ```
   PRINT (2 + 8 ) / 4 + 1
   ```

   Die Ausgabe sollte wie nachstehend aussehen:

   ```
    3.5
   ```

2. Geben Sie `CLS` im Direktfenster ein, und löschen Sie den Ausgabebildschirm mit Hilfe der EINGABETASTE.

Die **PRINT**-Anweisung ist sehr flexibel, sie kann jegliche Kombination von Zeichenketten, Zahlen, Variablen und Ausdrücken anzeigen. Die auszugebenden Elemente werden durch Semikolons (;) oder Kommata (,) getrennt. Sind zwei Elemente durch einen Semikolon getrennt, wird das zweite Element unmittelbar nach dem ersten angezeigt, wie aus nachstehendem Beispielprogramm ersichtlich:

1. Geben Sie folgende Zeile im Direktfenster ein, und betätigen Sie die EINGABETASTE:

   ```
   PRINT "Summe von 5.29 und"; 2.79; "ist"; 5.29 + 2.79
   ```

   Die Ausgabe erscheint wie folgt:

   ```
   Summe von 5.29 und 2.79 ist 8.08
   ```

2. Tippen Sie `CLS` im Direktfenster ein, und drücken Sie die EINGABETASTE, um den Ausgabebildschirm zu löschen.

Es folgt ein zweites Beispielprogramm zu diesem Vorgang:

1. Löschen Sie das Arbeitsfenster mit Hilfe des Befehls **Neues Programm** aus dem Menü **Datei**.
2. Geben Sie folgendes Beispiel im Arbeitsfenster ein:

   ```
   Zahl = 36
   PRINT "Die Summe von"; Zahl; "und 4 ist"; Zahl + 4
   PRINT "Die Quadratzahl von"; Zahl; "ist""; Zahl * Zahl
   ```

3. Wählen Sie den Befehl **Start** aus dem Menü **Ausführen** aus.

   Die Ausgabe erscheint wie folgt:

   ```
   Die Summe von 36 und 4 ist 40
   Die Quadratzahl von 36 ist 1296
   ```

4. Geben Sie `CLS` im Direktfenster ein, und löschen Sie den Ausgabebildschirm über die EINGABETASTE.

Sind zwei Elemente einer **PRINT**-Anweisung durch ein Komma getrennt, erscheint das zweite Element an dem nächsten festgelegten Tabulatorstopp. (Der Abstand zwischen Tabulatorstopps beträgt 14 Spalten.) Beachten Sie den Unterschied zwischen der Ausgabe der folgenden **PRINT**-Anweisung und der obenstehenden ähnlichen Ausgabe mit den Semikolons:

1. Geben Sie folgende Zeilen im Direktfenster ein, und betätigen Sie die EINGABETASTE:

   ```
   PRINT "Summe von 5,29 und", 2,79, "ist", 5,29 + 2,79
   ```

   Die Ausgabe erscheint wie folgt:

   ```
   Summe von 5,29 und          2,79          ist          8,08
   ```

2. Geben Sie `CLS` im Direktfenster ein, und löschen Sie den Ausgabebildschirm durch Betätigen der EINGABETASTE.

Kommata und Semikolons können in derselben **PRINT**-Anweisung zur gewünschten Positionierung der Elemente verwendet werden.

Wie aus nachstehendem Beispiel ersichtlich, druckt normalerweise jede **PRINT**-Anweisung die entsprechende Ausgabe in einer neuen Zeile:

1. Löschen Sie das Arbeitsfenster anhand des Befehls **Neues Programm** aus dem Menü **Datei**.
2. Geben Sie folgendes Beispiel im Arbeitsfenster ein:

   ```
   PRINT "Wie alt sind Sie?"
   PRINT 41
   ```

3. Wählen Sie den Befehl **Start** aus dem Menü **Ausführen** aus.

   Die Ausgabe sollte wie folgt aussehen:

   ```
   Wie alt sind Sie?
    41
   ```

4. Geben Sie `CLS` im Direktfenster ein, und löschen Sie den Ausgabebildschirm durch Betätigen der EINGABETASTE.

Wenn die Ausgabe mehrerer **PRINT**-Anweisungen auf derselben Zeile kombiniert werden soll, ist ein Semikolon (;) am Ende jeder **PRINT**-Anweisung einzusetzen. Das vorhergehende Beispielprogramm ist wie folgt zu ändern:

1. Löschen Sie das Arbeitsfenster mit Hilfe des Befehls **Neues Programm** aus dem Menü **Datei**.

2. Geben Sie folgendes Beispiel im Arbeitsfenster ein:

```
PRINT "Wie alt sind Sie?";
PRINT 41
```

3. Wählen Sie den Befehl **Start** aus dem Menü **Ausführen** aus:

   Die Ausgabe erscheint wie folgt:

```
Wie alt sind Sie? 41
```

4. Geben Sie `CLS` im Direktfenster ein, und löschen Sie den Ausgabebildschirm durch Betätigen der EINGABETASTE.

Dateneingabe anhand der INPUT-Anweisung

Die **INPUT**-Anweisung zeigt eine Zeichenkette zur Eingabeaufforderung an und wartet anschließend auf die Eingabe des Benutzers. Der als Antwort auf die Eingabeaufforderung eingegebene Wert wird der in der **INPUT**-Anweisung angegebenen Variablen zugewiesen.

```
INPUT "Geben Sie Ihr Alter ein: ", Alter%
PRINT Alter%
```

Wird `39` als Antwort auf die Eingabeaufforderung `Geben Sie Ihr Alter ein:` eingetippt, wird der Wert `39` der Variablen `Alter%` zugewiesen, und dieser Wert wird dann von der nächsten Anweisung `PRINT Alter%` ausgegeben.

INPUT nimmt auch Zeichenketten an:

```
INPUT "Wie heißen Sie? ", IhrName$
PRINT IhrName$
```

Eine Zeichenkettenvariable nimmt jede Eingabe an, da alle Einträge - sogar die Zahlen - Zeichenketten darstellen. (Die Eingabe `42` besteht in diesem Fall nicht aus der Zahl 42, sondern aus den Zeichen `4` und `2`. Wenn es sich um eine numerische Eingabevariable handelt, wandelt BASIC diese Zeichenkette in eine Zahl um. Ist die Eingabevariable eine Zeichenkette, wird die Zeichenfolge `42` einfach der Zeichenkette zugeordnet.)

Einer numerischen Variablen kann jedoch nur eine Zahl zugeordnet werden. Wenn Sie in diesem Fall anstelle einer Zahl das Wort `Hund` eingeben, antwortet BASIC mit der Fehlermeldung `Korrigieren Sie die Eingabe` und einer erneuten Eingabeaufforderung.

Datenfelder

Die Höchsttemperaturen für jeden Tag des Monats Juli können anhand von 31 Gleitkommavariablen mit dem Namen `Juli1, Juli2, Juli3`, usw. bis `Juli31` erfaßt werden.

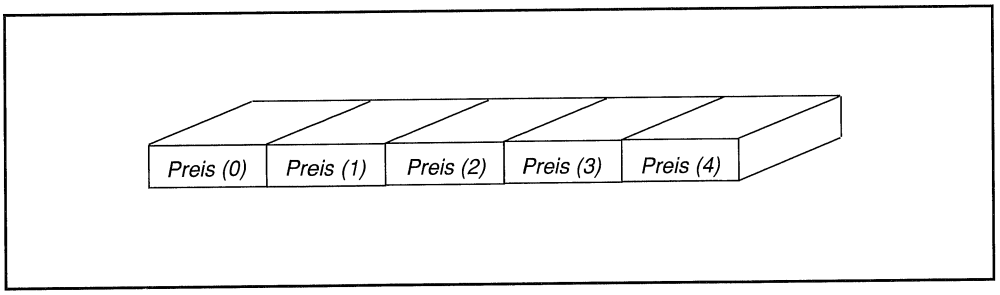
Eine Gruppe zusammenhängender, jedoch einzeln benannter Variablen ist aber schwierig zu handhaben, da jede Variable einzeln behandelt werden muß. Wenn Sie beispielsweise eine Eingabeaufforderung für die Temperatur jedes Julitages einsetzen möchten, müßten Sie 31 einzelne Anweisungen zur Eingabeaufforderung schreiben.

```
INPUT "Geben Sie die Temperatur für den 1. Juli ein ", Juli1
    .
    .
    .
INPUT "Geben Sie die Temperatur für den 31. Juli an ", Juli31
```

Ein ähnliches Problem tritt beim Versuch auf, ein Datenabrufprogramm zu schreiben. Es ist nicht zweckmäßig ein Programm zu schreiben, das alle 31 Variablennamen anzeigt, um den Wert eines bestimmten Tages (zum Beispiel die Höchsttemperatur für den 4. Juli - d.h. der Wert von `Juli4`) abzurufen. Es wäre erheblich bequemer, wenn der Computer nach Eingabe des Datums (die Zahl `4`) die richtige Variable und den korrekten Wert selbst auswählen könnte.

Die Lösung ist, dem gesamten Wertsatz denselben Namen (in diesem Fall `Juli`) zuzuweisen und die einzelnen Werte durch einen numerischen "Index" zu unterscheiden. Eine Gruppe identisch benannter Variablen, die durch deren Index unterschieden werden, bilden ein "Datenfeld" (siehe Abbildung 4.2 zur Verdeutlichung). Datenfelder sind besonders leistungsfähig in Verbindung mit Schleifen (siehe Abschnitt mit dem Titel "Wiederholung von Programmoperationen").

Abbildung 4.2 Anhand der BASIC-Anweisung DIM Preis (4) erstelltes Datenfeld



Deklarieren von Datenfelder

Eine Datenfeldvariable muß zuerst mit Hilfe einer **DIM**-Anweisung deklariert werden, die die Größe des Datenfeldes bestimmt. Vor dem Einsatz einer Datenfeldvariablen muß eine **DIM**-Anweisung stehen.

Der in der **DIM**-Anweisung angegebene Wert bestimmt den Höchstwert des Variablenindex. Folgende Anweisung deklariert zum Beispiel ein aus 32 Gleitkommavariablen mit dem Namen `Juli` bestehendes Datenfeld, dessen Indizes von 0 bis 31 verlaufen:

```
DIM Juli(31)
```

DIM läßt sich auch mit Zeichenkettenvariablen einsetzen:

```
DIM ZeichkeVar$(250)
```

Angabe der Datenfeldelemente

Der *n*-te Bestandteil (oder "Element") eines Datenfeldes kann durch den Einsatz der Zahl *n* (oder einer Variablen oder eines Ausdrucks mit dem Wert *n*) in Klammern nach dem Variablennamen angegeben werden. Auf diese Weise kann das Programm eine spezifische Variable mit der eingegebenen Nummer verbinden. Das vierte Element des Datenfeldes `Juli` kann zum Beispiel durch `Juli(4)`, durch `Juli(d%)` - wenn `d%` gleich vier ist - oder auch durch `Juli(SQR(16))` spezifiziert werden.

Nachstehender Code fordert den Benutzer auf, das Datum einzugeben und gibt dann die Höchsttemperatur für diesen Julitag aus (unter der Voraussetzung, daß alle Temperaturwerte dem Datenfeld `Juli` zugewiesen wurden).

```
DIM Juli(31)
INPUT "Geben Sie das Datum im Juli an: ", Datum%
PRINT "Höchsttemperatur am"; Datum%; "Juli war"; _
   Juli(Datum%)
```

Beim Fällen von Entscheidungen verwendete logische Vergleiche

BASIC kann entscheiden, ob zwei Zahlen oder Zeichenketten identisch oder verschieden sind. Aufgrund dieser Entscheidung kann eine entsprechende Gruppe von Programmanweisungen ausgeführt oder wiederholt werden.

Angenommen, der vom zu versteuernden Einkommen absetzbare Pauschalabzug beträgt 2.500 DM. Wenn die Summe der einzelnen absetzbaren Abzüge unter 2.500 DM liegt, wird der Pauschalbetrag abgesetzt. Wenn jedoch die Summe der einzelnen Abzüge den Pauschalbetrag überschreitet, werden diese einzeln aufgeführt. Ein Einkommenssteuerprogramm ist in der Lage, diese Entscheidung zu treffen und anschließend den erforderlichen Programmteil, der absetzbare Abzüge einzeln aufführt oder der den Pauschalbetrag absetzt, auszuführen, ohne daß der Benutzer die Aktion des Programms zu steuern braucht.

Folgende Abschnitte erklären, wie BASIC im Programm anwendbare logische Entscheidungen fällt.

Vergleichsoperatoren

Zahlen und Zeichenketten werden mit Hilfe von "Vergleichsoperatoren" verglichen. Die sechs möglichen Beziehungen werden durch folgende Operatoren dargestellt:

Vergleichsoperator	*Bedeutung*
=	Gleich
<>	Ungleich
>	Größer als
<	Kleiner als
>=	Größer gleich
<=	Kleiner gleich

Hinweis BASIC benutzt das Gleichheitszeichen (=) sowohl beim Zuordnen eines Wertes einer Variablen als auch bei der Angabe der logischen Gleichheit und erkennt die Bedeutung des Zeichens aus dem Kontext.

Folgende Liste verdeutlicht die Auswertung dieser Beziehungen:

Vergleichsoperator	*Auswertung*
`7 = 35`	Falsch
`7 <> 6`	Wahr
`6 > 1`	Wahr
`4 < 3`	Falsch
`7 <= 7`	Wahr
`8 >= 7`	Wahr
`Äpfel = Orangen`	Hängt von den Variablenwerten ab

Was die letzte Anweisung anbetrifft, sind Äpfel natürlich keine Orangen. Wenn jedoch `Äpfel` und `Orangen` Variablen bezeichnen, hängt die Richtigkeit der Anweisung von dem Wert der Variablen ab.

Es stellt sich die Frage, ob die von den nachstehenden Vergleichsoperatoren angestellten Zeichenkettenvergleiche wahr oder falsch sind.

```
"Tarzan" <> "Jane"
"Y" = "Y"
"Y" = "y"
"fragen">"fraglich"
```

Da es sich bei `Tarzan` und `Jane` um verschiedene Zeichenketten handelt, ist die Behauptung wahr. `Y` ist identisch mit `Y`, so daß auch die zweite Behauptung wahr ist. `Y` und `y` sind jedoch nicht identisch und somit ist die dritte Behauptung falsch.

Die letzte Anweisung kann verwirrend sein, da auf den ersten Blick eine Zeichenkette kaum "größer", oder "kleiner" als eine andere ist. BASIC fällt jedoch die Entscheidung durch den schrittweisen Vergleich jedes einzelnen Zeichens, bis es Zeichen antrifft, die nicht übereinstimmen. In diesem Beispiel stimmen die Zeichenketten ab dem fünften Zeichen nicht mehr überein. Die Entscheidung über "größer" oder "kleiner" wird dann auf diese unterschiedlichen Zeichen wie folgt gegründet.

Innerhalb des Computers werden Zeichen durch Zahlen dargestellt, die der Zeichenreihenfolge im Alphabet entsprechen. Somit weisen beispielsweise am Ende des Alphabets stehende Buchstaben einen höheren numerischen Wert als die vorhergehenden Buchstaben auf. Da `l` im Alphabet nach `e` steht, ist `l` ein "größerer" Buchstabe und die Anweisung ist demnach falsch. `Fraglich` ist "größer" als `fragen`. Die Fähigkeit, den relativen Wert einer Zeichenkette zu bestimmen, erleichtert das Schreiben eines Programms, das eine Liste von Wörtern oder Namen alphabetisch anordnet.

In den meisten Beispielen wurden Zahlen mit Zahlen und Zeichenketten mit Zeichenketten verglichen. Es ist jedoch möglich, beliebige Kombinationen von Zahlen und numerischen Variablen oder von Zeichenketten und Zeichenkettenvariablen untereinander zu vergleichen. Nur numerische Werte und Zeichenkettenwerte können nicht untereinander verglichen werden, da der Vergleich vom logischen Standpunkt belangslos ist.

Boolesche Ausdrücke

Logische Behauptungen werden als "boolesche Ausdrücke" bezeichnet, nach George Boole, der einige der Regeln der mathematischen Logik formuliert hat. Boolesche Operationen haben als einziges Ziel die Feststellung, ob ein boolescher Ausdruck wahr oder falsch ist. Zum Unterschied von mathematischen Ausdrücken, die zahlenmäßig auswerten, geben boolesche Ausdrücke lediglich einen der zwei Werte aus: "wahr" oder "falsch".

Computer stellen "wahr" und "falsch" anhand von Zahlen dar. Die BASIC-Codierung von "wahr" und "falsch" ist aus Eingabe und Ausführung folgender zwei Zeilen zu erkennen:

1. Löschen Sie das Arbeitsfenster anhand des Befehls **Neues Programm** aus dem Menü **Datei**.
2. Geben Sie folgendes Beispiel im Arbeitsfenster ein:

```
PRINT 5 > 6
PRINT 6 > 5
```

3. Wählen Sie den Befehl **Start** aus dem Menü **Ausführen** aus.

   Die Ausgabe erscheint wie folgt:

```
 0
-1
```

   In der Anzeige ist `0` in der ersten und `-1` in der zweiten Zeile zu sehen.

   Die Zahl `5` ist nicht größer als `6`, und deshalb gibt BASIC 0, den Wert für "falsch" aus. Die Zahl `6` ist größer als `5`, und BASIC gibt -1, den Wert für "wahr" aus.

4. Geben Sie `CLS` im Direktfenster ein und löschen Sie den Ausgabebildschirm durch Betätigen der EINGABETASTE.

Zusammengesetzte Ausdrücke

In BASIC ergibt sich ein zusammengesetzter boolescher Ausdruck aus der Verbindung zweier boolescher Ausdrücke mit einem "logischen Operator". Die am häufigsten verwendeten logischen Operatoren sind **AND** und **OR**.

Der **AND**-Operator erfordert, daß beide Ausdrücke wahr sind, wenn der zusammengesetzte Ausdruck wahr sein soll. Beim Einsatz des **OR**-Operators braucht nur einer der Ausdrücke wahr zu sein, damit der gesamte Ausdruck wahr ist.

Die folgenden Beispiele bestehen aus zusammengesetzten Ausdrücken, die zwei einfache Ausdrücke mit Hilfe von **AND** und **OR** verbinden:

Ausdruck	*Auswertung*
`10 > 5 AND  100 < 200`	Wahr
`3 < 6 AND 7 > 10`	Falsch
`8 < 7 OR 90 > 80`	Wahr
`2 < 1 OR 3 > 60`	Falsch
`"J" > "N" AND "Ja" <>` "Nein"	Wahr
`"Y" < "N" OR 4 <> 4`	Falsch

Der **NOT**-Operator kehrt den wahren oder falschen Wert eines Ausdruckes in das Gegenteil um:

Ausdruck	*Auswertung*
`NOT (5 > 10)`	Wahr
`NOT (8 < 7 OR 90 > 80)`	Falsch
`NOT (3 < 6 AND 7 > 10)`	Wahr
`NOT (0)`	Wahr

Die Logik des letzten Beispiels ist nicht unbedingt offensichtlich. In BASIC bedeutet falsch Null. Demnach bedeutet `NOT (0)` "nicht falsch", also wahr. Mit Hilfe der **AND**-, **OR**- und **NOT**-Kombinationen lassen sich Ausdrücke aufbauen, deren Komplexität praktisch nur von der Lesbarkeit eines Ausdrucks beschränkt ist.

Bei der Auswertung boolescher Ausdrücke wird der Vorrang in derselben Reihenfolge wie bei den mathematischen Ausdrücken angenommen, wobei **AND** der Multiplikation und **OR** der Addition entsprechen. Die Auswertung wird von links nach rechts vorgenommen und beginnt mit den einfachen Ausdrücken. Anschließend werden die mit **AND** verknüpften Ausdrücke und zuletzt die mit **OR** verknüpften Ausdrücke ausgewertet. Wie bei arithmetischen Ausdrücken, kann die Reihenfolge der Auswertung ebenfalls durch Klammern gesteuert werden, anstatt lediglich auf die Vorrangregeln zu achten.

Weitere Erläuterungen zu dem BASIC-Wert -1 für wahr sind Kapitel 1, "Strukturen zur Ablaufsteuerung", des Handbuches *Programmieren in BASIC* zu entnehmen.

Die nächsten Abschnitte beschreiben die Anwendung boolescher Ausdrücke in Programmen.

Verwendung logischer Anweisungen zur Ablaufsteuerung des Programms

Der Mensch entscheidet seine nächste Tätigkeit aufgrund seiner Wünsche und augenblicklichen Prioritäten. BASIC verfügt über einen ähnlichen Mechanismus, der dem Programm die Auswahl der Operationen bzw. des auszuführenden Programmteils erlaubt.

Dieser Mechanismus besteht aus der Anweisung **IF...THEN...ELSE**. Wenn ein boolescher Ausdruck wahr ist, werden die nach **THEN** stehenden Anweisungen ausgeführt. Ist ein Ausdruck falsch, werden die nach **ELSE** stehenden Anweisungen ausgeführt. Ein boolescher Ausdruck muß entweder wahr oder falsch sein, so daß eine der Anweisungsgruppen ausgeführt wird. Beide Anweisungsgruppen können nicht ausgeführt werden.

Der Vorgang, im Laufe dessen das Programm eine der zwei oder mehreren möglichen Aktionen (oder Abschnitte des Programmcodes) auswählt, wird "Verzweigen" genannt. Die Analogie des Begriffes bezieht sich auf einen Baumast, der in mehrere Äste verzweigt.

Nachstehend finden Sie die Syntax der Anweisung **IF...THEN...ELSE**:

IF *boolescherAusdruck* **THEN**
führen Anweisungen eine Aktion aus
ELSE
führen Anweisungen eine andere Aktion aus
END IF

Setzen Sie im nächsten einfachen Beispiel die Anweisung **IF...THEN...ELSE** ein, um zu entscheiden, ob eine eingegebene Zahl größer, kleiner oder gleich 100 ist:

1. Löschen Sie das Arbeitsfenster mit Hilfe des Befehls **Neues Programm** aus dem Menü **Datei**.
2. Geben Sie folgendes Beispiel im Arbeitsfenster ein:

```
INPUT "Geben Sie eine Zahl ein: ", n
IF n > 100 THEN
   PRINT n; "ist größer als 100"
ELSE
   PRINT n; "ist kleiner oder gleich 100"
END IF
```

3. Wählen Sie den Befehl **Start** aus dem Menü **Ausführen** aus.

   Das logische Gegenstück von "größer" ist "kleiner oder gleich" (nicht "kleiner"). Führen Sie dieses Beispiel mit mehreren *n*-Werten, einschließlich dem Wert 100 aus, um zu überprüfen, daß die Anweisung `kleiner oder gleich 100` angezeigt wird.
4. Geben Sie `CLS` im Direktfenster ein, und löschen Sie den Ausgabebildschirm mit Hilfe der EINGABETASTE.

Eingehende Erläuterungen zu der Anweisung **IF...THEN...ELSE** und anderen in QuickBASIC vorhandenen Entscheidungsstrukturen sind Kapitel 1, "Strukturen zur Ablaufsteuerung", des Handbuches *Programmieren in BASIC* zu entnehmen.

Wiederholung von Programmoperationen

BASIC bietet mehrere Möglichkeiten zur wiederholten Ausführung einer Gruppe von Programmanweisungen. Diese können sooft wie angegeben wiederholt werden, oder bis eine bestimmte logische Bedingung wahr wird. Soll ein Anweisungsblock 100 Mal ausgeführt werden, braucht dieser nur einmal eingegeben zu werden, da die BASIC-Strukturen zur Ablaufsteuerung die Wiederholungen steuern.

Die Fähigkeit, Programmanweisungen zu wiederholen ist vielfach anwendbar. Um beispielsweise eine Zahl *n* zur Potenz *p* zu erheben, muß *n p* mal mit sich selbst multipliziert werden. In einem 10-teiligen Datenfeld muß der Benutzer 10 Mal zur Dateneingabe aufgefordert werden.

Die FOR...NEXT-Schleife

Die **FOR...NEXT**-Schleife bildet eine der Möglichkeiten zur Wiederholung eines Programmabschnitts. Der Programmanweisungsblock zwischen den Schlüsselwörtern **FOR** und **NEXT** wird, wie im nächsten Beispiel gezeigt, sooft wie angegeben ausgeführt.

1. Löschen Sie das Arbeitsfenster anhand des Befehls **Neues Programm** aus dem Menü **Datei**.
2. Geben Sie folgendes Beispiel im Arbeitsfenster ein:

```
FOR Zähler% = 1 TO 5
   PRINT "diese Zeile wird 5 Mal ausgegeben;";
   PRINT " das ist Wiederholung"; Zähler%
NEXT Zähler%
```

3. Wählen Sie den Befehl **Start** aus dem Menü **Ausführen** aus.

   Die Ausgabe erscheint wie folgt:

```
diese Zeile wird 5 Mal ausgegeben; das ist Wiederholung 1
diese Zeile wird 5 Mal ausgegeben, das ist Wiederholung 2
diese Zeile wird 5 Mal ausgegeben, das ist Wiederholung 3
diese Zeile wird 5 Mal ausgegeben, das ist Wiederholung 4
diese Zeile wird 5 Mal ausgegeben, das ist Wiederholung 5
```

4. Geben Sie `CLS` im Direktfenster ein, und löschen Sie den Ausgabebildschirm über die EINGABETASTE.

Im oben aufgeführten Beispiel wird die Variable `Zähler%` als "Schleifenvariable" oder "Schleifenzähler" benutzt. Die zwei, durch das Schlüsselwort **TO** getrennten, nach dem Gleichheitszeichen stehenden Zahlen stellen den Anfangs- bzw. Endwert des Schleifenzählers dar. In diesem Beispiel ist 1 der Anfangswert und 5 der Endwert.

Vor der ersten Ausführung der Anweisungen in der **FOR...NEXT**-Schleife, wird dem Schleifenzähler ein Anfangswert zugewiesen (in diesem Fall `1`). Nach jeder Ausführung der Schleifenanweisungen, erhöht BASIC die Variable `Zähler%` automatisch um eine Zahl. Dieser Zyklus wird fortgesetzt, bis `Zähler%` größer ist als der Endwert (in diesem Fall `5`). Demnach werden die zwischen den Schlüsselwörtern **FOR** und **NEXT** stehenden Anweisungen in diesem Beispiel fünfmal ausgeführt.

Um eine **FOR...NEXT**-Schleife *n* Mal zu wiederholen, müßten die Anfangs- und Endwerte normalerweise 1 bzw. *n* betragen. Diese Werte sind jedoch nicht unbedingt erforderlich. Es können auch Werte benutzt werden, die sich voneinander durch *n* - 1 unterscheiden. Die Werte -2 und 2 würden zum Beispiel ebenfalls fünf Wiederholungen veranlassen.

Der Zähler kann die Indexvariable eines Datenfeldes darstellen. (Die **FOR...NEXT**-Schleife wird oft für den Zugriff auf jedes Element eines Datenfeldes verwendet.) Im nächsten Beispiel wird der Benutzer aufgefordert, die Höchsttemperaturen für jeden Julitag einzugeben:

```
DIM Juli!(31)
FOR Datum% = 1 TO 31
   PRINT "Geben Sie die Höchsttemperatur für den";Datum%;
   INPUT "Juli ein"; Juli!(Datum%)
NEXT Datum%
```

Weder der Anfangs- noch der Endwert müssen eine Konstante sein, einer oder beide können Variablen sein. Das folgende Beispiel erfordert Anfangs- und Endwerte und gibt dann die Quadratwurzel jedes Wertes aus dem betreffenden Bereich aus:

```
INPUT "Anfangswert? ", AnfangsWert%
INPUT "Endwert? ", EndWert%
FOR Zähler% = AnfangsWert% TO EndWert%
   PRINT "die Quadratwurzel von"; Zähler%; "ist";_
SQR(Zähler%)
NEXT Zähler%
```

Die Inkrementvorgabe beträgt 1, falls nicht anders angegeben. Die Anweisung **STEP** setzt einen unterschiedlichen Inkrementwert, wie aus folgender Prozedur ersichtlich:

1. Löschen Sie das Arbeitsfenster mit Hilfe des Befehls **Neues Programm** aus dem Menü **Datei**.

2. Geben Sie im Arbeitsfenster folgendes Beispiel ein:

```
FOR Zähler% = 5 TO 1 STEP -1
   PRINT "diese Zeile wird 5 Mal ausgegeben;";
   PRINT "das ist Wiederholung"; Zähler%
NEXT Zähler%
```

3. Wählen Sie den Befehl **Start** aus dem Menü **Ausführen** aus.

   Die Ausgabe sieht wie folgt aus:

```
diese Zeile wird 5 Mal ausgegeben; das ist Wiederholung 5
diese Zeile wird 5 Mal ausgegeben; das ist Wiederholung 4
diese Zeile wird 5 Mal ausgegeben; das ist Wiederholung 3
diese Zeile wird 5 Mal ausgegeben; das ist Wiederholung 2
diese Zeile wird 5 Mal ausgegeben; das ist Wiederholung 1
```

4. Geben Sie `CLS` im Direktfenster ein, und drücken Sie die EINGABETASTE, um den Ausgabebildschirm zu löschen.

Das Vorzeichen des Schrittwerts muß mit dem Anfangs- und dem Endwert übereinstimmen. Wenn der Endwert größer als der Anfangswert ist, muß der Schrittwert positiv sein. Ist dagegen der Endwert kleiner als der Anfangswert, muß der Schritt negativ sein. Wenn der Schritt mit dem falschen Vorzeichen versehen ist, wird die **FOR...NEXT**-Schleife überhaupt nicht ausgeführt, sondern einfach übersprungen.

Die DO...LOOP-Schleife

Bis jetzt wurde nur eine Möglichkeit zur Wiederholung eines Programmabschnitts untersucht, die **FOR....NEXT**-Schleife, die jedoch sehr begrenzt ist. Obwohl der Anfangs- bzw. Endpunkt angegeben wird, ist die Schleife selbst festgelegt, das heißt, daß sie bedingungslos sooft wie angegeben ausgeführt wird.

Diese Einschränkung hat in dem Beispiel, in dem der Benutzer zur Eingabe der 31 Höchsttemperaturen im Juli aufgefordert wurde, keine Probleme aufgeworfen, da der Monat Juli immer 31 Tage hat. Es kann jedoch vorkommen, daß die Anzahl der erforderlichen Wiederholungen einer Berechnung oder Operation nicht bekannt ist.

Wenn Sie beispielsweise in jeder Zeile eines Textes nach einem bestimmten Wort suchen, ohne zu wissen, auf welcher Zeile es sich befindet (andernfalls wäre die Suche überflüssig), können Sie die Anzahl der Wiederholungen einer **FOR...NEXT**-Schleife nicht angeben. In diesem Fall ist eine flexible, nicht festgelegte Möglichkeit zur Wiederholung von Programmanweisungsgruppen angebracht.

DO...LOOP stellt diesen flexibleren Wiederholungsmechanismus dar. Die zwischen den Schlüsselwörtern **DO** und **LOOP** stehenden Anweisungen werden, wie im nächsten Beispiel gezeigt, unbegrenzt wiederholt:

1. Löschen Sie das Arbeitsfenster mit Hilfe des Befehls **Neues Programm** aus dem Menü **Datei**.
2. Geben Sie folgendes Beispiel im Arbeitsfenster ein:

```
DO
   PRINT "Sich im Kreise drehen..."
LOOP
```

3. Wählen Sie den Befehl **Start** aus dem Menü **Ausführen** aus.

   `Sich im Kreise drehen...` wird fortwährend wiederholt. (Beenden Sie das Programm durch Betätigen der Tastenkombination STRG+UNTBR.)
4. Geben Sie `CLS` im Direktfenster ein, und löschen Sie den Ausgabebildschirm durch Betätigen der EINGABETASTE.

Die oben aufgeführte einfache **DO...LOOP**-Schleife ist endlos. BASIC hat jedoch die Möglichkeit, die Schleife zum richtigen Zeitpunkt zu beenden, wenn die logische Bedingung zum Schleifenabschluß angegeben wird.

Zu diesem Zweck kann die **WHILE**-Bedingung verwendet werden, die den Computer anweist, mit der Ausführung der Schleife fortzufahren, solange die Bedingung wahr *ist*. Eine andere Möglichkeit besteht in der Anwendung der Bedingung **UNTIL**, die die Ausführung der Schleife veranlaßt, bis eine Bedingung wahr *wird*.

Bei einer **DO WHILE**-Schleife muß die Bedingung am Anfang wahr sein, und die Schleife wird ausgeführt, solange diese Bedingung wahr bleibt. Bei einer **DO UNTIL**-Schleife ist die Bedingung am Anfang falsch, und die Schleife wird ausgeführt, bis die Bedingung wahr wird.

Die benutzerdefinierte Bedingung bildet einen booleschen Ausdruck. Dieser Ausdruck vergleicht meistens eine Variable mit einer Konstanten. Erfüllt der Ausdruck die vom Programmierer festgelegte logische Bedingung (wenn beispielsweise die Variable gleich der Konstanten wird), wird die Schleife beendet. (Jeder Vergleich ist möglich, aber der Vergleich zwischen einer Schleifenvariablen und einer Zahl oder einer Zeichenkette ist am häufigsten.)

Die DO WHILE LOOP-Schleife

Eine Schleife **DO WHILE LOOP** sieht wie folgt aus:

DO WHILE *booleschерAusdruck*
  zu wiederholende Anweisungen
LOOP

Nachstehend wird eine einfache **DO WHILE**-Schleife vorgestellt, die die Arbeitsweise dieses Schleifentyps verdeutlicht. Die Abschlußbedingung der Schleife ist die Gleichheit zwischen `groß!` und `klein!`.

```
groß! = 256.0
klein! = 1.0
DO WHILE groß! <> klein!
   PRINT "groß ="; groß!; "  klein ="; klein!
   groß! = groß! / 2.0
   klein! = klein! * 2.0
LOOP
```

Die zwei Variablen `groß!` und `klein!` werden außerhalb der Schleife initialisiert. Innerhalb der Schleife werden die Variablenwerte zuerst ausgegeben, und dann wird `groß!` durch 2 dividiert und `klein!` mit 2 multipliziert. Der Vorgang wird solange wiederholt, bis beide gleich 16 sind, und die Schleife beendet wird.

Die Anfangswerte von `groß!` und `klein!` wurden so ausgewählt, daß sie am Ende übereinstimmten. Andernfalls würde sich die Schleife unbegrenzt wiederholen. (In Wirklichkeit hält das Programm letztlich unabhängig von den ausgewählten Werten an. Der Wert der Variablen `klein!` wird durch wiederholte Verdoppelung so groß, daß er durch eine BASIC-Gleitkommavariable nicht mehr dargestellt werden kann. Das Programm hält an diesem Punkt an und gibt `eine Überlauf`-Fehlermeldung aus.)

Es ist zu beachten, daß die Ausführung einer **DO WHILE**-Schleife einen wahren booleschen Ausdruck erfordert. Sind die Anfangswerte von `groß!` und `klein!` identisch, ist der Ausdruck `groß! <> klein!` falsch, wird die Schleife überhaupt nicht ausgeführt, sondern übersprungen. Vergewissern Sie sich immer vor Eingabe einer Schleife, daß die Variablen des booleschen Ausdrucks die gewünschten Werte haben.

Die DO UNTIL-Schleife

Eine **DO UNTIL**-Schleife ist formmäßig identisch mit einer **DO WHILE**-Schleife, mit dem Unterschied, daß Erstere das Schlüsselwort **UNTIL** verwendet.

DO UNTIL *booleschерAusdruck*
  zu wiederholende Ausdrücke
LOOP

Die im vorhergehenden Beispiel gezeigte Schleife kann auch wie nachstehend unter Verwendung des Schlüsselwortes **UNTIL** geschrieben werden:

```
groß! = 256,0
klein! = 1,0
DO UNTIL groß! = klein!
   PRINT "groß ="; groß!; "  klein ="; klein!
   groß! = groß! / 2,0
   klein! = klein! * 2,0
LOOP
```

Eine **DO UNTIL**-Schleife wird fortgesetzt, bis ein boolescher Ausdruck wahr wird, so daß eine andere Abschlußbedingung erforderlich ist. In diesem Fall handelt es sich um das Gegenstück der im vorhergehenden Beispiel aufgeführten Bedingung: `groß! = klein!`.

Es folgt ein praktischeres Beispiel einer **DO...LOOP**-Schleife. Der Benutzer wird wiederholt aufgefordert, Ja oder Nein (abgekürzt durch `J` oder `N`) einzugeben, bis diese Eingabe erfolgt:

```
DO UNTIL InZeichenkette$ = "Y" OR InZeichenkette$ = "N"
   PRINT "Geben Sie Ja oder Nein (J/N) ein: ";
   INPUT "";InZeichenkette$
LOOP
PRINT "Sie haben eingegeben"; InZeichenkette$
```

Es ist zu beachten, daß `j` und `n` nicht akzeptiert werden, da `InZeichenkette$` nur mit Großbuchstaben verglichen wird.

Hinweis In den angegebenen Schleifenbeispielen werden sowohl **WHILE**- als auch **UNTIL**-Anweisungen nach **DO** benutzt. Es ist ebenfalls möglich, diese nach dem Schlüsselwort **LOOP** einzusetzen. In diesem Fall werden zuerst die Anweisungen in der Schleife ausgeführt und anschließend die Bedingung getestet. Diese Vorgehensweise stellt sicher, daß die Schleife mindestens einmal ausgeführt wird. Dieses Verfahren wird in Kapitel 1, "Strukturen zur Ablaufsteuerung", des Handbuches *Programmieren in BASIC* erläutert.

Schreiben eines einfachen BASIC-Programms

Sie haben die Grundbegriffe der Programmiertätigkeit in BASIC anhand der Beispiele kennengelernt und sind jetzt in der Lage, das nachstehend beschriebene einfache Programm zu schreiben, um Ihre Kenntnisse in die Praxis umzusetzen.

Das Programm berechnet den Durchschnitt mehrerer Zahlen und könnte in folgender Reihenfolge geschrieben werden:

1. Fordern Sie den Benutzer auf, die Anzahl der Zahlen einzugeben, deren Durchschnitt berechnet werden soll.
2. Wenn die Anzahl Null oder negativ ist, geben Sie lediglich eine Warnmeldung aus.
3. Wenn die Anzahl positiv ist, wählen Sie einen Variablennamen für die laufende Summe, und setzen Sie diesen auf Null.
4. Fordern Sie zur Zahleneingabe einzeln auf. Teilen Sie dem Benutzer die laufende Nummer des jeweils eingegebenen Werts mit (1. Zahl, 2. Zahl, usw.).
5. Nach Eingabe aller Zahlen führen Sie die Berechnung durch, und geben Sie den Durchschnittswert aus.

Dieses Programm verwendet nur die in diesem Kapitel beschriebenen BASIC-Sprachelemente und erfordert keine "Programmiertricks". Nachstehend wird auch eine Lösung vorgestellt, es empfiehlt sich jedoch, das eigene Programm vor Vergleich mit dieser Lösung zu schreiben. Es handelt sich allerdings um ein "Nachschlageprojekt". Lesen Sie, falls erforderlich, dieses Kapitel noch einmal durch, um die gewünschten Anweisungen und die entsprechenden Erläuterungen zu finden.

Ein Programm kann auf unterschiedlicher Art und Weise geschrieben werden. Wenn das eigene Programm funktioniert, ist es richtig, auch wenn es vom nachstehenden Programm abweicht.

```
PRINT "Geben Sie die Anzahl der Zahlen an, deren"; INPUT "
Durchschnitt berechnet werden soll ", Anzahl%
IF Anzahl% <= 0 THEN
   PRINT "Keine gültige Anzahl; muß größer als 0 sein."
ELSE
   Summe! = 0.0          ' Setze laufende Summe auf Null.
   FOR Zähler% = 1 TO Anzahl%
      PRINT "Zahl"; Zähler%
      INPUT " ", Wert!
      Summe! = Summe! + Wert!     ' Addiere nächsten
                                  ' Wert zur Summe.
   NEXT Zähler%
   PRINT "Der Durchschnittswert ist"; Summe! / Anzahl%
END IF
```

Weitere Informationen

Sie sind jetzt in der Lage, einfache Programme in BASIC zu schreiben, haben jedoch noch sehr viel über das Programmieren in BASIC zu lernen.

Die QuickBASIC-Handbücher und die On-Line-Hilfe bieten vollständige Informationen über den Gebrauch der QuickBASIC-Umgebung und über sämtliche QuickBASIC-Anweisungen und Funktionen. Die Dokumentation umfaßt jedoch keine komplette Einführung in die BASIC-Sprache und erläutert keine allgemeinen Programmiergrundsätze oder die Verwendung des Betriebssystems DOS. Die nachstehend zu Ihrer Information aufgeführten Bücher umfassen ausführlichere Erläuterungen zu diesen Themen. Mit Ausnahme der eigenen Veröffentlichungen beurteilt Microsoft diese Bücher in keiner Weise und zieht sie anderen Publikationen über dieselben Themen nicht vor.

BASIC-Bücher

Baloui, Said. *Effektives Programmieren mit QuickBASIC*. Markt & Technik Verlag, Haar 1988.

Craig, John Clark. *Microsoft QuickBASIC Programmer's Toolbox*. Redmond, Wash.: Microsoft Press, 1988.

Dwyer, Thomas A. und Critchfield, Margot. *BASIC and the Personal Computer*. Reading, Mass.: Addison-Wesley Publishing Co., 1978.

Enders, Bernd und Petersen, Bob. *BASIC Primer for the IBM PC & XT*. New York, N.Y.: New American Library, 1984.

Feldman, Phil und Rugg, Tom. *Using QuickBASIC 4*. Carmel, Ind.: Que Corporation, 1988.

Hergert, Douglas. *Microsoft QuickBASIC*. Microsoft Press, Vieweg Verlag, Wiesbaden 1988.

Inman, Don und Albrecht, Bob. *Using QuickBASIC*. Berkeley, Cal.: Osborne McGraw-Hill, 1988.

Bücher über DOS

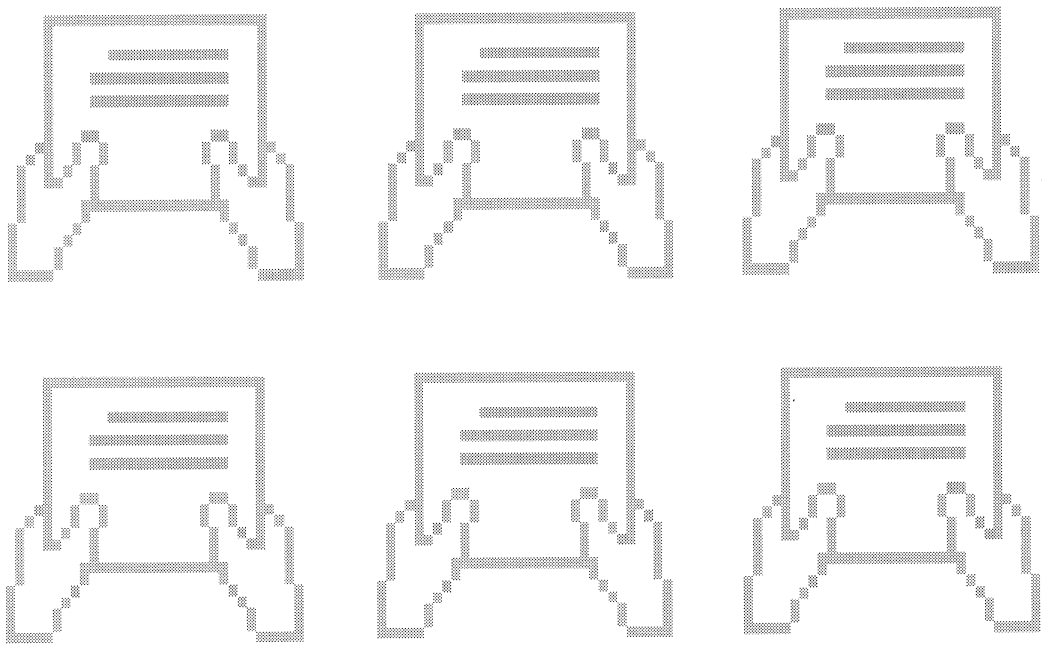
Duncan, Ray. *MS-DOS für Fortgeschrittene*. Microsoft Press, Vieweg Verlag, Wiesbaden 1987.

Wolverton, Van. *MS-DOS*. Microsoft Press, Vieweg Verlag, Wiesbaden 1987.

Wolverton, Van. *MS-DOS Aufbaukurs*. Microsoft Press, Vieweg Verlag, Wiesbaden 1987.

2. Teil: Programmentwicklung in der QuickBASIC-Umgebung

Q 2. Teil



Programmentwicklung in der QuickBASIC-Umgebung

Der 2. Teil, "Programmentwicklung in der QuickBASIC-Umgebung", zeigt dem Benutzer am Beispiel von *qkarten.bas* wie man ein reelles Anwendungsprogramm entwickelt. Obwohl *qkarten.bas* ein gutes Beispiel für strukturiertes Programmieren ist, zielt es in erster Linie darauf hinaus, in die QuickBASIC-Programmierumgebung einzuführen, nicht die Programmiertechniken selbst zu beschreiben. Die zu den jeweiligen Codeblocks gehörenden Kommentare tragen zum Verständnis der Programme bei, aber Sie können die einzelnen Kapitel durchgehen, ohne die im vorhergehenden Abschnitt angewandten Programmierkonzepte in allen Einzelheiten verstanden zu haben. Das Lehrmaterial hilft Ihnen, drei sehr komplexe QuickBASIC-Einrichtungen – den intelligenten Editor, die On-Line-Hilfe und die Fehlerbeseitigung – in wirklichkeitsgetreuen Programmiersituationen kennenzulernen. Anschließend können Sie die Programmkommentare und den Code durchlesen und die On-Line-Hilfe, sowie die Debug-Einrichtungen von QuickBASIC anwenden, um die Funktionsweise jedes Blocks zu verstehen.

Kapitel 5 beschreibt die Struktur des Programms QKARTEN, sowie seine Benutzeroberfläche und erklärt die Eingabe einer **SUB...END SUB**-Prozedur. Kapitel 6 zeigt, der intelligente Editor während der Code-Eingabe bei der Verfolgung und Korrektur von Programmierfehlern behilflich ist. Kapitel 7 und 8 bieten Ihnen die Gelegenheit, die On-Line-Hilfe beim Schreiben eigener Programme zu verwenden. In Kapitel 9 wird ein Programmfehler verfolgt und verbessert, um die korrekte Ausführung von QKARTEN sicherzustellen. Am Ende des 9. Kapitels ist QKARTEN funktionell und einsatzbereit.

Kapitelverzeichnis

5 Das Programm QKARTEN
6 Bearbeiten im Arbeitsfenster
7 Programmieren mit der On-Line-Hilfe
8 Verwendung des Beispielcodes aus der On-Line-Hilfe
9 Fehlerbeseitigung während der Programmierung

5 Das Programm QKARTEN

Es wird *qkarten.bas* beschrieben, ein Programm, das den nächsten Kapiteln als Grundlage dient. Sie werden es mit Code in Form einer "Prozedur" ergänzen. Eine Prozedur ist eine Routine, die, einmal geschrieben, zur Durchführung einer bestimmten Aufgabe von verschiedenen Stellen des Programms aufgerufen werden kann. In diesem Kapitel werden Sie folgende Verfahren kennenlernen:

- Laden eines Programms beim Aufruf von QuickBASIC.
- Durchgehen des Programmtexts von *qkarten.bas*.
- Aufrufen von Prozeduren aus dem Direktfenster.
- Erstellen und Aufrufen einer **SUB**-Prozedur.

Dieses Kapitel erfordert ungefähr eine Arbeitsstunde.

Der Aufbau des Programms QKARTEN

Während des Durcharbeitens der Kapitel 5 bis 9 werden Sie das Programm QKARTEN mit Programmzeilen ergänzen. Die Codeeingaben, die zum Abschluß von QKARTEN nötig sind, erscheinen unter eigenen Überschriften. Sie können Ihre Arbeit nach Abschluß jeder dieser durchnumerierten Eingaben speichern. Falls Sie während einer Eingabe einen Fehler machen, brauchen Sie lediglich die zuvor gespeicherte Version erneut zu laden und die Eingabe zu wiederholen.

Hinweis Beim Durchlesen des Programms QKARTEN werden Sie eventuell einige Codefehler entdecken. Diese Fehler wurden absichtlich eingebaut, um sie während der Arbeit an QKARTEN zu verbessern.

Laden eines Programms beim Starten von QuickBASIC

Durch Eingabe des Programmnamens als "Argument" auf der Befehlszeile, kann mit dem Aufruf von QuickBASIC gleichzeitig ein Programm geladen werden. Der Begriff "Argument" bezeichnet die einem Befehl erteilte Information, die zur Modifikation des Befehls führt.

Hinweis Falls Ihnen keine Festplatte zur Verfügung steht, kopieren Sie die Dateien *qkarten.bas* und *qkarten.dat* auf eine leere Diskette und verwenden diese um das Programm fertigzustellen.

◊ Geben Sie folgende Zeile nach der DOS-Eingabeaufforderung ein, um QuickBASIC zu starten und gleichzeitig *qkarten.bas* zu laden:

```
qb qkarten
```

Nach dieser Eingabe sucht QuickBASIC im aktuellen Arbeitsverzeichnis nach einer Datei mit dem Basisnamen *qkarten* und der Erweiterung *.bas*. Wenn es die Datei findet, wird diese geladen. Andernfalls nimmt QuickBASIC an, daß Sie mit dem Schreiben eines Programms mit dem Namen *qkarten.bas* beginnen möchten.

QKARTEN – Ein kurzer Überblick

Wenn Sie das Lernprogramm QB-Express abgeschlossen haben, sind Ihnen einige Teile von QKARTEN bereits vertraut. Sie werden die Gelegenheit haben, in den Kapiteln 5 bis 9 tatsächlich mit dem QKARTEN-Code zu arbeiten.

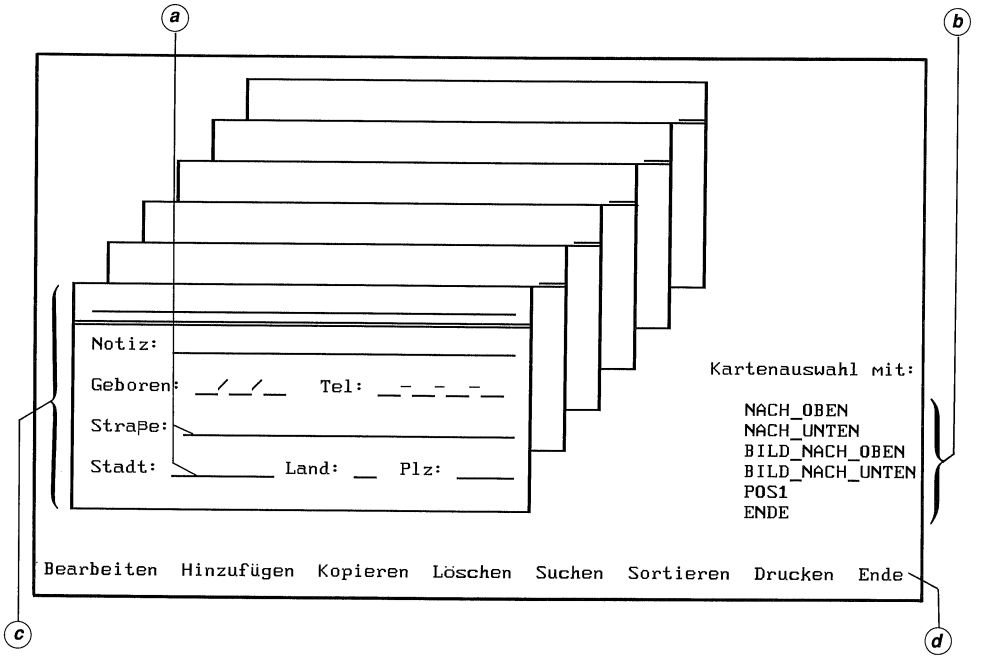
Als Programmieranfänger mag Ihnen dieses Material zum Teil schwierig vorkommen, da es sich bei QKARTEN um ein reelles Anwendungsprogramm handelt. Machen Sie sich jedoch darüber keine Gedanken, da Sie die Kenntnisse eines Abschnitts vor Durchlesen des nächsten Abschnitts nicht zu beherrschen brauchen. Dieses Handbuch ist zum Kennenlernen der zur QuickBASIC-Umgebung gehörenden Dienstprogramme bestimmt und soll nicht als Lehrbuch zum Erlernen von Programmiertechniken dienen.

Auch wenn Sie bereits in BASIC programmiert haben, haben Sie die Möglichkeit, einige besonders leistungsfähige Techniken anhand eines strukturierten Programms wie QKARTEN, das häufig Prozeduren verwendet, kennenzulernen. Falls Sie BASIC-Übersetzer, wie GW-BASIC® oder BASICA, benutzt haben, wird Sie dieses Programm angenehm überraschen.

Das Programm QKARTEN

QKARTEN ist ein Datenbankprogramm. Es erleichtert die Arbeit mit kleinen, als "Datensätze" bezeichneten Informationseinheiten, die in einer Datei gespeichert sind. QKARTEN verwaltet die Datei, indem es diese öffnet oder schließt und die Datensätze beim Ändern, Ergänzen oder Löschen bestimmter Informationen oder ganzer Datensätze aktualisiert. Der interessanteste Teil des Programms QKARTEN ist die "Benutzeroberfläche", anhand der der Benutzer mit QKARTEN arbeitet (siehe Abbildung 5.1).

Abbildung 5.1 Die QKARTEN-Benutzeroberfläche



a) Datensätze bestehen aus Feldern, wie Straße und Stadt.
b) Die QKARTEN-Hilfe.
c) Jede Karte stellt einen Datensatz dar.
d) Die QKARTEN-Befehlszeile.

QKARTEN stellt die Daten mit Hilfe des Prinzips einer Indexkartendatei in einer dem Benutzer vertrauten Weise dar. Jede auf dem Bildschirm zu sehende Karte entspricht einem Datensatz der Datenbank. Die Karten bestehen aus mehreren "Feldern" (separate Kartenbereiche), mit deren Hilfe die Eingabe neuer Daten sowie die Änderung und Umordnung bestehender Daten erfolgt.

Nachdem Sie QKARTEN kennengelernt haben, werden Sie in der Lage sein, Daten (und das Programm selbst) zu ändern und es zum Speichern der gewünschten Information zu benutzen. QKARTEN ist zwar ein umfangreiches Programm, aber nach strukturierten Programmiergrundsätzen organisiert, mit anderen Worten, teilt es komplizierte Aufgaben in mehrere einfache auf. Dies ermöglicht das schrittweise Erlernen der entsprechenden Programmiertechniken. Beim Lesen der nächsten Abschnitte empfiehlt es sich, den jeweils beschriebenen Teil des Programms QKARTEN auf dem Bildschirm durchzugehen.

Vereinbarungen und Definitionen

QKARTEN beginnt mit einem kurzen Abschnitt, der das Programmziel beschreibt, und wird mit einem Abschnitt über Vereinbarungen und Definitionen fortgesetzt. Die Vereinbarungen und Definitionen verwenden Anweisungen wie **DEFINT**, **CONST**, **TYPE...END TYPE**, **DECLARE**, **DIM** und **REDIM**, die QuickBASIC erlauben, den für die Programmvariablen benötigten Speicherplatz abzuschätzen. Beispielsweise weist `DEFINT A-Z` QuickBASIC an, alle Variablen, für die nicht explizit ein Datentyp deklariert wurde, als Ganzzahl zu betrachten.

CONST-Anweisungen verbessern die Lesbarkeit des Programms, indem sie Werten, die vom Computer nur als Zahlen erkannt werden, beschreibende Namen zuweisen. Der Wert `0` (Null) stellt zum Beispiel in QuickBASIC die logische Bedingung "falsch" dar. In Zeile 45 definiert eine **CONST**-Anweisung die symbolische Konstante `FALSCH` als den Wert Null. Diese Definition ermöglicht die Darstellung des Wertes `0` durch das Wort `FALSCH` in logischen Vergleichen.

TYPE...END TYPE-Konstruktionen deklarieren "Aggregatdatentypen", die sowohl Zeichenketten als auch alle Arten von numerischen Werten umfassen können. Zum Beispiel können nach Deklarieren des Datentyps `PERSON` in Zeile 64 Variablen von diesem benutzerdefinierten Typ definiert werden.

Die mit einer **DECLARE**-Anweisung beginnenden Zeilen erlauben QuickBASIC die Überprüfung von Prozeduraufrufen, um deren korrekte Anwendung sicherzustellen.

Kommentare

Den Vereinbarungen und Definitionen folgt ein längerer Abschnitt, der zum größten Teil Kommentare beinhaltet. Jede Kommentargruppe beschreibt einen Block ausführbarer, in QKARTEN einzugebende Anweisungen, die die allgemeine Ablaufsteuerung des Programms definieren. Dieser QKARTEN-Abschnitt endet mit einer **END**-Anweisung, die das Ende der normalen Ausführungsfolge des Programms kennzeichnet.

Nach der Anweisung END stehende Anweisungen

Der abschließende Programmabschnitt beginnt nach der **END**-Anweisung. Vor jeder Anweisungsgruppe dieses Abschnitts steht eine "Marke", die aus einem Namen und einem Doppelpunkt besteht. (Die QuickBASIC-Marken gleichen den Zeilennummern in BASIC-Interpretern.) Die mit diesen Marken zusammenhängenden Anweisungen werden nur dann ausgeführt, wenn zuvor andere Anweisungen, die auf die betreffenden Marken Bezug nehmen, ausgeführt werden. Die erste Marke (`SpeicherFehl`) kennzeichnet eine Fehlerbehandlungsroutine. Beim Auftreten eines bestimmten Fehlertyps während der Ausführung des Programms QKARTEN zeigen die nach `SpeicherFehl` stehenden Anweisungen eine Fehlermeldung auf dem Bildschirm an und beenden das Programm. Die restlichen markierten Abschnitte enthalten **DATA**-Anweisungen, die bei der Ausgabe der Indexkarten auf den Bildschirm ausgeführt werden.

Aufrufen der QKARTEN-Prozeduren aus dem Direktfenster

QKARTEN weist zwei Programmierebenen auf. Die Modulebene enthält die allgemeine Programmlogik, während die Prozedurebene zahlreiche einzelne Prozeduren umfaßt, die die Kleinarbeit des Programms durchführen.

Im Direktfenster können sowohl die Auswirkungen einer BASIC-Anweisung betrachtet als auch Programmprozeduren aufgerufen werden. Auf diese Weise kann man die Auswirkungen einer Prozedur sehen, ohne das gesamte Programm auszuführen. Jede Prozedur führt eine bestimmte Aufgabe durch. Nachstehende Arbeitsschritte veranschaulichen den Aufruf von Prozeduren, die die QKARTEN-Benutzeroberfläche auf dem Ausgabebildschirm anzeigen:

1. Bringen Sie den Cursor durch Betätigen der Taste F6 in das Direktfenster.

2. Geben Sie `CLS` ein, und drücken Sie die EINGABETASTE, um den Ausgabebildschirm zu löschen. Betätigen Sie eine beliebige Taste, um in das Direktfenster zurückzukehren.
3. Geben Sie `CALL ZeichKart` ein, und drücken Sie die EINGABETASTE.

   QuickBASIC führt jetzt die Prozedur `ZeichKart` aus. Ein Prozeduraufruf ähnelt der Ausführung einer BASIC-Anweisung. Nachdem die Prozedur die jeweilige Aufgabe durchgeführt hat, wartet QuickBASIC auf die nächsten Instruktionen.
4. Kehren Sie durch Drücken einer Taste in das Direktfenster zurück, und geben Sie folgendes genau wie angegeben in die nächste Zeile des Direktfensters ein:

   ```
   CALL ZeigBefzeile  : DO : LOOP
   ```
5. Führen Sie die eingegebene Zeile durch Betätigen der EINGABETASTE aus.

   Die Prozedur `ZeigBefzeile` fügt die QKARTEN-Befehlszeile den bereits angezeigten Teilen hinzu. Zu diesem Zeitpunkt zeigt QuickBASIC jedoch nicht sofort die Eingabeaufforderung

   ```
   Beliebige Taste zum Fortsetzen drücken
   ```

   an, da die nach der Prozedur `ZeigBefzeile` stehenden `DO`- und `LOOP`-Anweisungen eine bedingungslose (endlose) Schleife erzeugen.

Unterbrechen einer bedingungslosen Schleife in QKARTEN

Beim Erstellen von QKARTEN werden Sie bedingunglose Schleifen unterbrechen müssen, da das Programm auf Betätigen der Tasten noch nicht reagiert. QuickBASIC sucht jedoch nach einer bestimmten Tastenkombination, STRG+UNTBR, die es Ihnen erlaubt, eine endlose Schleife zu unterbrechen und in die QuickBASIC-Umgebung zurückzukehren. Beachten Sie bitte, daß der Benutzer nach Einfügen des gesamten Codes in QKARTEN in der Lage ist, Befehle durch Drücken der auf der QKARTEN-Befehlszeile hervorgehobenen Buchstaben auszuführen. Die Q-TASTE wird beispielsweise das Programm beenden und die Kontrolle an QuickBASIC zurückgeben, da QKARTEN jede der vom Benutzer betätigten Tasten überprüft und die erforderliche Aktion durchführt. Eine bedingungslose Schleife wird wie folgt unterbrochen:

◊ Geben Sie die Kontrolle an QuickBASIC durch Betätigen der Tastenkombination STRG+UNTBR zurück.

Wie Sie feststellen können, kehrt der Cursor bei Betätigen der Tastenkombination STRG+UNTBR nicht in das Direktfenster zurück, in dem die letzte ausgeführte Anweisung steht, sondern ins Arbeitsfenster.

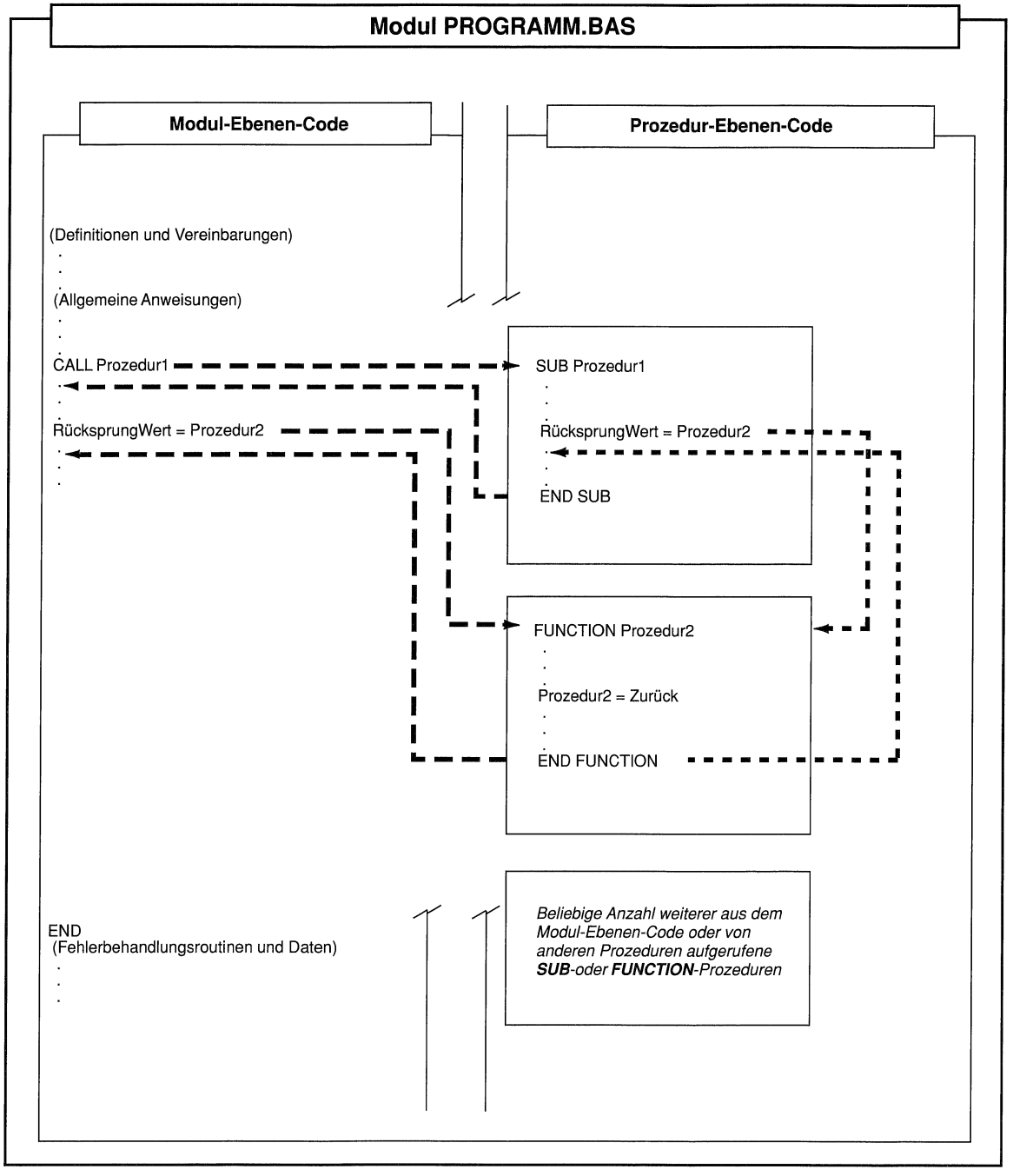
Der Modul-Ebenen-Code

Jedes QuickBASIC-Programm weist eine Modul-Ebene auf, die beim Öffnen des Programms im Arbeitsfenster erscheint. Der größte Teil des in QKARTEN einzufügenden Codes wird als "Modul-Ebenen-Code" bezeichnet und führt die in den Kommentaren beschriebenen Programmaufgaben anhand der (mit QKARTEN-Prozeduren verbundenen) BASIC-Anweisungen und -Funktionen durch. Der eingefügte Modul-Ebenen-Code definiert, wie das Programm QKARTEN mit dem Benutzer in Verbindung steht und Informationen austauscht.

Strukturierte Programmierung mit Prozeduren

Es können auch Programme nur auf Modul-Ebene geschrieben werden. Dies erschwert es jedoch, das Programm lesbar zu gestalten und Fehler zu beseitigen. Um dies zu erleichtern, überträgt QKARTEN die meisten Aufgaben an Prozeduren wie `ZeichKart` und `ZeigBefzeile`, die vor Durchführung der entsprechenden Aufgaben aufgerufen werden. Abbildung 5.2 veranschaulicht die Beziehung zwischen Modul-Ebenen-Code, Prozeduren und deren jeweiligen Ausführungsfolgen.

Abbildung 5.2 Module und Prozeduren



Eine Prozedur besteht meistens aus einer Anweisungsgruppe, die eine einzige Aufgabe durchführt. Prozeduren werden in derselben Datei wie der Modul-Ebenen-Code gespeichert, wobei QuickBASIC Prozeduren jedoch als eigenständige, vom Modul-Ebenen-Code und voneinander getrennte Dateneinheiten behandelt. Jedes Programm verfügt über eine Modul-Ebene, während eine Prozedur-Ebene erst nach Definition einer Prozedur existiert. QuickBASIC führt die Anweisungen einer Prozedur nur beim expliziten Aufruf der jeweiligen Prozedur aus. Prozeduren können aus dem Modul-Ebenen-Code, aus anderen Prozeduren oder aus dem Direktfenster aufgerufen werden.

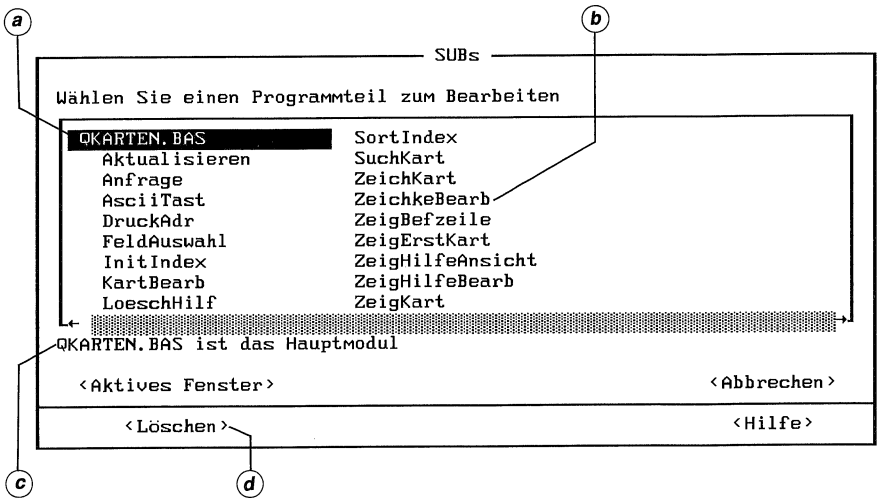
Die Struktur des Programms

Obwohl sie sich in derselben Datei befinden, trennt QuickBASIC die Prozeduren vom Modul-Ebenen-Code und den anderen Prozeduren. Aus diesem Grund können eine Prozedur und der Modul-Ebenen-Code (oder zwei Prozeduren) nicht gleichzeitig in demselben Fenster erscheinen. Um die unterschiedlichen Programmteile zu bearbeiten, müssen sie mit Hilfe des Befehls **SUBs** aus dem Menü **Ansicht** in das Arbeitsfenster gebracht oder daraus entfernt werden. Das Dialogfeld **SUBs** führt sämtliche Prozeduren eines Programms auf. Sehen Sie sich alle Bestandteile von QKARTEN wie folgt an:

1. Drücken Sie ALT+N und S, um den Befehl **SUBs** aus dem Menü **Ansicht** auszuwählen.

   Das Dialogfeld **SUBs** ermöglicht den Zugriff auf alle Programmteile (siehe Abbildung 5.3). Beim ersten Eintrag, *qkarten.bas*, handelt es sich um einen Dateinamen, der den Modul-Ebenen-Code des Programms darstellt.

Abbildung 5.3 Das Dialogfeld SUBs



a) Name des Moduls, das die darunterstehenden Prozeduren enthält.

b) Die Prozeduren stehen eingerückt unter dem Modulnamen.

c) Beschreibt das markierte Element.

d) Löscht eine Datei aus einem Programm oder eine Prozedur aus einer Datei.

2. Markieren Sie den ersten Eintrag unter *qkarten.bas* durch Betätigen der NACH UNTEN TASTE.

Unmittelbar unter *qkarten.bas* steht der Eintrag `Aktualisieren`. Jeder eingerückte, unter *qkarten.bas* stehende Eintrag stellt eine vom Programm verwendete **SUB**- oder **FUNCTION**-Prozedur dar. Sämtliche Prozeduren werden in der Datei *qkarten.bas* gespeichert.

3. Wählen Sie die Prozedur `ZeichKart` aus, und drücken Sie die EINGABETASTE.

   QuickBASIC zeigt den Code der Prozedur `ZeichKart` im Arbeitsfenster an. In der Titelleiste des Arbeitsfensters ist jetzt

   `QKARTEN.BAS:ZeichKart`

   zu sehen. Wenn Sie die Prozedur `ZeichKart` anhand der RICHTUNGSTASTEN durchgehen, werden Sie feststellen, daß sich kein anderer Text im Arbeitsfenster befindet.

4. Drücken Sie ALT+N und S, für den Befehl **SUBs** aus dem Menü **Ansicht**, und wählen Sie *qkarten.bas*, falls erforderlich, aus.

5. Laden Sie den Modul-Ebenen-Code durch Betätigen der EINGABETASTE wieder in das Arbeitsfenster.

In QuickBASIC können zwei Prozedurtypen definiert werden: **SUB**- und **FUNCTION**-Prozeduren. Um den Code einer Prozedur in der Ausführungsfolge des Programms einzufügen, ist deren Bezeichnung an der Stelle einzusetzen, an der die entsprechende Aufgabe durchgeführt werden soll. Eine **SUB**-Prozedur läßt sich durch Einsetzen ihres Namens in einer **CALL**-Anweisung aufrufen. **FUNCTION**-Prozeduren werden wie die internen QuickBASIC-Standardfunktionen (zum Beispiel **TIME$**) angewandt. Prozeduren bieten eine wirksame Möglichkeit zur Codierung häufig durchgeführter Aufgaben.

Definieren einer Prozedur in QKARTEN

Prozeduren werden am besten durch Schreiben einfacher Prozeduren erlernt. Die folgenden Arbeitsschritte zeigen, wie QuickBASIC jede einzelne Prozedur vom Rest des Programms automatisch trennt und wie eine einfache, später in QKARTEN verwendete Prozedur definiert und aufgerufen wird.

1. Code-Eingabe in QKARTEN

1. Positionieren Sie den Cursor in einer Leerzeile am Ende des Vereinbarungs- und Definitionsabschnitts im Arbeitsfenster (ungefähr Zeile 102).

2. Erstellen Sie eine weitere Leerzeile durch Betätigen der EINGABETASTE.
3. Geben Sie die Worte `SUB Alarm` ein, und drücken Sie die EINGABETASTE.

   QuickBASIC öffnet ein neues Fenster und beginnt die Prozedur mit den Anweisungen `SUB Alarm` und `END SUB`. Die Anweisung `DEFINT A-Z` erscheint über der `SUB`-Anweisung, da die erstellte Prozedur den Standard-Datentyp des Modul-Ebenen-Codes übernimmt.

   In der Titelleiste des Arbeitsfensters erscheint jetzt der Name des Moduls (`QKARTEN.BAS`) und `Alarm`, der Name der im Arbeitsfenster vorhandenen Prozedur.
4. Geben Sie folgende Kommentare und Code-Blocks genau wie angegeben zwischen den Anweisungen `SUB` und `END SUB` ein:

```
' Die Alarm-Prozedur sendet mit Hilfe der
' Anweisung SOUND Signale an den Lautsprecher des
' Computers und gibt ein Alarmsignal aus.
'
'
'
' Parameter: keine
'
' Ausgabe: Gibt dem Benutzer ein Alarmsignal
' Ändern Sie die Zahlen, um den Ton zu variieren.
FOR Ton = 600 TO 2000 STEP 40
   SOUND Ton, Ton / 7000
NEXT Ton
```

   Die Prozedur und ihre Auswirkung kann durch deren Aufruf aus dem Direktfenster getestet werden.
5. Bringen Sie den Cursor in das Direktfenster, geben Sie `CALL Alarm` ein und drücken Sie die EINGABETASTE.

Speichern des bearbeiteten Textes

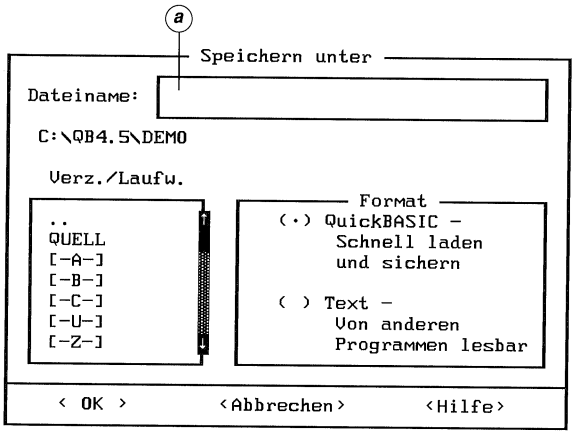
Von nun an werden Sie das Programm unter dem Namen des aktuellen Kapitels abspeichern. Wenn das Programm unter einem anderen Namen gespeichert wird, bleibt die alte Version (in diesem Fall *qkarten.bas*) unverändert auf der Diskette. Falls die nachfolgende Programmversion unbrauchbar ist, kann die vorhergehende Version geladen und neu bearbeitet werden.

2. Code-Eingabe in QKARTEN

1. Drücken Sie ALT+D und U, um den Befehl **Speichern unter...** aus dem Menü **Datei** auszuwählen.

   Das Dialogfeld **Speichern unter** erscheint auf dem Bildschirm (siehe Abbildung 5.4).

Abbildung 5.4 Das Dialogfeld Speichern unter...



a) Geben Sie den Namen des Programms an dieser Stelle ein.

2. Geben Sie den Namen `Kapitel5` im Textfeld ein, und betätigen Sie die EINGABETASTE.

   Durch dieses Verfahren wird das Programm *qkarten.bas* unter dem Namen *kapitel5.bas* auf der Diskette gespeichert. Die neu erstellte Prozedur `Alarm` wird als Bestandteil von *kapitel5.bas* mitgespeichert.

Weitere Informationen

Weitere Informationen über die in diesem Kapitel behandelten Themen sind folgenden Kapiteln zu entnehmen:

Kapitel	*Information*
Kapitel 11, "Das Menü Datei"	Vollständige Informationen über die Dateiverwaltung in QuickBASIC, einschließlich der Optionen zur Dateiformatierung und der Sonderbefehle zum Öffnen von verschiedenen Dateitypen, die nicht zu den Einzeldatei-Programmen gehören.
Kapitel 13, "Das Menü Bearbeiten", und Kapitel 14, "Das Menü Ansicht"	Vollständige Informationen zum Erstellen und Bearbeiten der QuickBASIC-Prozeduren.
Kapitel 2, "Prozeduren: Unterprogramme und Funktionen", des Handbuches *Programmieren in BASIC*	Erläuterungen der BASIC-Regeln zur Erstellung aller Arten von **SUB**- und **FUNCTION**-Prozeduren.

6 Bearbeiten im Arbeitsfenster

Die QuickBASIC-Editiereigenschaften ermöglichen dem Programmierer, Programme schnell, problemlos und genau zu schreiben und zu ändern. Während der Bearbeitung überprüft QuickBASIC den Code, um eine gültige BASIC-Syntax sicherzustellen. In diesem Kapitel werden Sie die Editierwerkzeuge zur Durchführung folgender Aufgaben einsetzen:

- Kennenlernen des intelligenten QuickBASIC-Editors.
- Verstehen der automatischen Textformatierung von QuickBASIC.
- Sofortiges Beantworten von Fragen mit Hilfe der On-Line-Hilfe.
- Einsetzen der Befehle **Ausschneiden**, **Kopieren** und **Einfügen** aus dem Menü **Bearbeiten**.
- Suchen und Ersetzen von Textpassagen.

Sie werden auch Modul-Ebenen-Code in QKARTEN eingeben. Für Fragen bezüglich der eingegebenen Wörter rufen Sie die On-Line-Hilfe auf, indem Sie den Cursor auf das Wort positionieren und F1 drücken. Die On-Line-Hilfe umfaßt vollständige Beschreibungen und Beispiele zu sämtlichen QuickBASIC-Anweisungen und -Funktionen. Beim Betätigen der Taste F1 erteilt QuickBASIC auch Informationen zu jeder an der Cursorstelle vorhandenen Variable, Prozedur oder symbolischen Konstante. Das Hilfsfenster läßt sich durch Betätigen der ESC-TASTE schließen.

Dieses Kapitel erfordert zwei Arbeitsstunden.

Der intelligente Editor

Der Bearbeitungsvorgang in QuickBASIC unterscheidet sich von dem eines gewöhnlichen Textverarbeitungssystems in erster Linie durch den "intelligenten Editor". QuickBASIC erkennt Schlüsselwörter der Programmiersprache und gibt eine Warnung aus, falls diese nicht korrekt verwendet werden. QuickBASIC fügt auch bestimmte Textarten selbständig im Programm ein. Beim Erstellen der Prozedur `Alarm` hat QuickBASIC zum Beispiel die Anweisungen `DEFINT A-Z` und `END SUB` nach Eingabe der Anweisung `SUB Alarm` selbständig eingesetzt.

Das automatische Formatieren

Der intelligente QuickBASIC-Editor schreibt Schlüsselworte automatisch groß und setzt Leerstellen zwischen Operatoren (wie Plus- und Minuszeichen) und den entsprechenden Operanden (den zu verarbeitenden Werten) ein. Wie aus folgenden Arbeitsschritten ersichtlich, verbessert diese Eigenschaft die Lesbarkeit des Programms:

3. Code-Eingabe in QKARTEN

1. Starten Sie QuickBASIC, und laden Sie *kapitel5.bas*, falls erforderlich.
2. Bringen Sie den Cursor mit Hilfe der Tasten BILD NACH UNTEN und NACH UNTEN in die Leerzeile unterhalb des nachstehenden Kommentars (ungefähr Zeile 106):

   ```
   ' Öffne Daten-Datei QKARTEN.DAT für Direktzugriff mit
   ' Hilfe von Datei # 1
   ```

3. Erstellen Sie eine Leerzeile mit der EINGABETASTE, und geben Sie folgende Anweisung genau wie angegeben, einschließlich des einleitenden Kommentarzeichens (des vor `open` stehenden Apostrophs), ein:

   ```
   ' open "qkarten.dat" for random as #1 len=len(Karte)
   ```

4. Drücken Sie die EINGABETASTE.

   QuickBASIC liest die Zeile wie einen Kommentar, so daß keine Formatierung durchgeführt wird.

5. Positionieren Sie den Cursor auf das Kommentarzeichen durch Betätigen der NACH OBEN TASTE.
6. Löschen Sie das Kommentarzeichen mit Hilfe der ENTF-TASTE.
7. Drücken Sie die NACH UNTEN TASTE.

   QuickBASIC wandelt die Schlüsselwörter (`OPEN`, `FOR`, `RANDOM`, `AS` und `LEN`) in Großbuchstaben um und fügt Leerstellen auf beiden Seiten des Gleichheitszeichens (=) ein.

Da die Eingabe von Kleinbuchstaben einfacher ist, werden die Schlüsselwörter der Codezeilen in diesem Buch kleingeschrieben. Wenn Sie nach deren Eingabe die EINGABETASTE betätigen, schreibt QuickBASIC die Schlüsselwörter automatisch groß. Worte, die keine Schlüsselwörter darstellen (wie symbolische Konstanten, Prozedur- und Variablennamen) erscheinen in gemischter Schreibweise oder in Großbuchstaben.

Die BASIC-Sprache "achtet nicht auf Groß- und Kleinschreibung", das bedeutet, daß sie zwei Versionen desselben Wortes anhand der Schreibweise nicht unterscheidet. Das Schlüsselwort `OPEN` bedeutet zum Beispiel in der BASIC-Sprache dasselbe wie `open` oder `OpEn`. Der intelligente Editor schreibt Schlüsselwörter nur konventionshalber groß. Der Variablenname `Karte` belegt denselben Speicherplatz wie `KArte` oder `KaRTe`.

Der intelligente Editor ist automatisch konsequent in der Großschreibung bestimmter Wörter eines Programmlistings. Nachstehende Liste faßt die in diesem Handbuch und in der Behandlung des eingegebenen Textes durch QuickBASIC verwendeten Konventionen zusammen:

Wortart	*Erscheinungsbild im Handbuch*	*QuickBASIC-Aktion*
BASIC-Schlüsselwörter	Kleinbuchstaben	Schreibt diese bei Eingabe der Zeile automatisch groß.
Variablen- und Prozedurnamen	Gemischte Schreibweise	Ändert alle Vorkommen des Namens gemäß der zuletzt angetroffenen Schreibweise um.
Symbolische Konstanten	Großbuchstaben	Ändert alle Vorkommen dieses Namens gemäß der zuletzt angetroffenen Schreibweise um.

Folgende Arbeitsschritte veranschaulichen die oben aufgeführten Regeln:

4. Code-Eingabe in QKARTEN

1. Bringen Sie den Cursor in die Zeile unterhalb des folgenden Kommentars (ungefähr Zeile 113):

```
' Zum Zählen der Datensätze einer Datei teile die Länge
' der Datei durch die Länge eines einzelnen Datensatzes;
' verwende die ganzzahlige Division (\) anstelle der
' normalen Division (/). Weise den resultierenden Wert
' LetztKart zu.
```

2. Schaffen Sie sich Freiraum mit Hilfe der EINGABETASTE, und geben Sie folgende Zeile genau wie angegeben ein:

```
LetztKart=LoF(1)\Len(Karte)
```

   Es ist zu beachten, daß in BASIC der rückwärtige Schrägstrich-Operator (\) eine ganzzahlige Division anstelle einer normalen Division durchführt.

3. Drücken Sie die EINGABETASTE.

   Die Schlüsselwörter werden großgeschrieben und die Leerstellen eingefügt. Die vorhergehende Schreibweise von `karte` (in der `OPEN`-Anweisung) wird gemäß der neuen Eingabe, `Karte`, aktualisiert. QuickBASIC geht von der Voraussetzung aus, daß die letzte Eingabe im Arbeitsfenster der gewünschten Schreibweise entspricht. Aus diesem Grund werden *alle* Vorkommen des Wortes, mit Ausnahme der Vorkommen in Kommentaren, geändert.

Syntaxüberprüfung

Sobald Sie eine durch Betätigen der EINGABETASTE oder einer RICHTUNGSTASTE bearbeitete Codezeile verlassen, überprüft QuickBASIC deren Syntax. Wenn die Zeile keinen gültigen BASIC-Code enthält, zeigt QuickBASIC ein Dialogfeld mit der Fehlerbeschreibung an.

Folgende Schritte zeigen, wie der intelligente Editor den Code auf Syntaxfehler prüft:

5. Code-Eingabe in QKARTEN

1. Bringen Sie den Cursor in die Zeile unterhalb des folgenden Kommentarblocks (ungefähr Zeile 130):

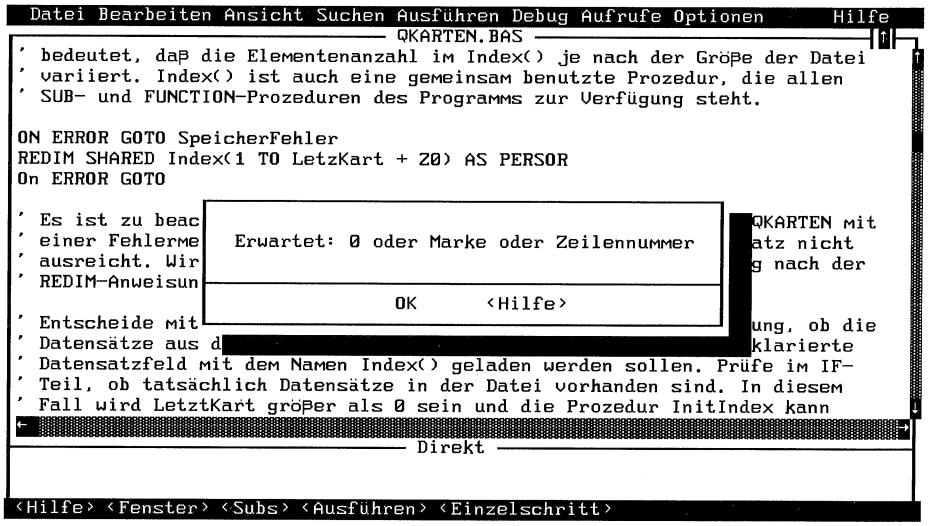
```
' Definiere das Index-Datenfeld neu, um die Datenfelder
' der Datei und 20 zusätzliche Datenfelder zu speichern,
' die dem Benutzer das Hinzufügen von Karten erlauben. Es
' handelt sich um ein dynamisches Datenfeld - das
' bedeutet, daß die Elementanzahl im Index() je nach der
' Größe der Datei variiert. Index() ist auch eine
' gemeinsam benutzte Prozedur, die allen SUB- und
' FUNCTION-Prozeduren des Programms zur Verfügung steht.
'
' Es ist zu beachten, daß eine Fehlerverfolgung das
' Beenden von QKARTEN mit einer Fehlermeldung veranlaßt,
' falls der verfügbare Speicherplatz nicht ausreicht.
' Wird kein Fehler entdeckt, wird die Fehlerverfolgung
' nach der REDIM-Anweisung ausgeschaltet.
```

2. Betätigen Sie die EINGABETASTE, und geben Sie folgende Zeilen genau wie angegeben, einschließlich der Fehler ein:

```
on error goto SpeicherFehl
redim shared Index(1 to LetztKart + 20) as PERSOR
on error goto
```

3. Drücken Sie die EINGABETASTE nach der Abschlußzeile. QuickBASIC gibt folgende Fehlermeldung aus (siehe Abbildung 6.1):

Abbildung 6.1 Das Dialogfeld Fehlermeldung



Wenn die Fehlerverfolgung wie in diesem Beispiel anhand von **ON ERROR GOTO** ausgeschaltet wird, erwartet QuickBASIC anschließend die Zahl 0.

4. Löschen Sie das Dialogfeld durch Betätigen der ESC-TASTE.

   QuickBASIC positioniert den Cursor an der Stelle, an der der Fehler entdeckt wurde.

5. Fügen Sie eine Null am Ende der Zeile hinzu:

   ```
   ON ERROR GOTO 0
   ```

Nach Schließen eines Dialogfeldes für Fehlermeldungen überprüft QuickBASIC die Syntax der Anweisung nicht mehr, bis die Zeile wieder geändert wird. Falls die Zeile nicht mehr bearbeitet wird, erinnert QuickBASIC erst während des Programmablaufs an den noch bestehenden Fehler.

Während des Programmablaufs entdeckte Fehler

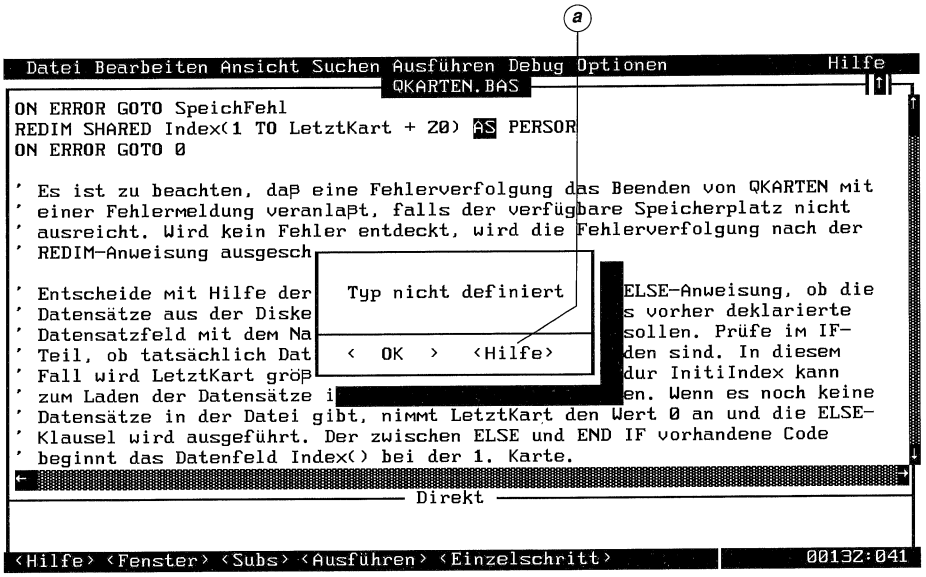
Einige Fehler treten erst bei der Code-Ausführung auf. In der `REDIM`-Anweisung wurde beispielsweise das Datenfeld `Index()` als zum Typ `PERSOR` gehörend deklariert. Der Name des benutzerdefinierten Typs `PERSON`, den Sie in Kapitel 5, "Das Programm QKARTEN", Vereinbarungs- und Definitionsabschnitt des Programms angetroffen haben, wurde hier falsch geschrieben. In der Annahme, daß der in keiner Vereinbarung angetroffene Typ `PERSOR` irgendwo bereits deklariert wurde, hat QuickBASIC diesen Fehler bei Eingabe nicht entdeckt.

Nachstehender Vorgang verdeutlicht die Entdeckung dieser Fehler:

◊ Drücken Sie ALT+A und S, um den Befehl **Start** aus dem Menü **Ausführen** auszuwählen und die eingegebenen Zeilen auszuführen.

QuickBASIC gibt die Fehlermeldung `Typ nicht definiert` aus und hebt das Wort `AS` aus der Anweisung `REDIM` wie in Abbildung 6.2 hervor.

Abbildung 6.2 Fehlermeldung



a) Wählen Sie diese Befehlsfläche, um Hilfe für eine Fehlermeldung anzufordern.

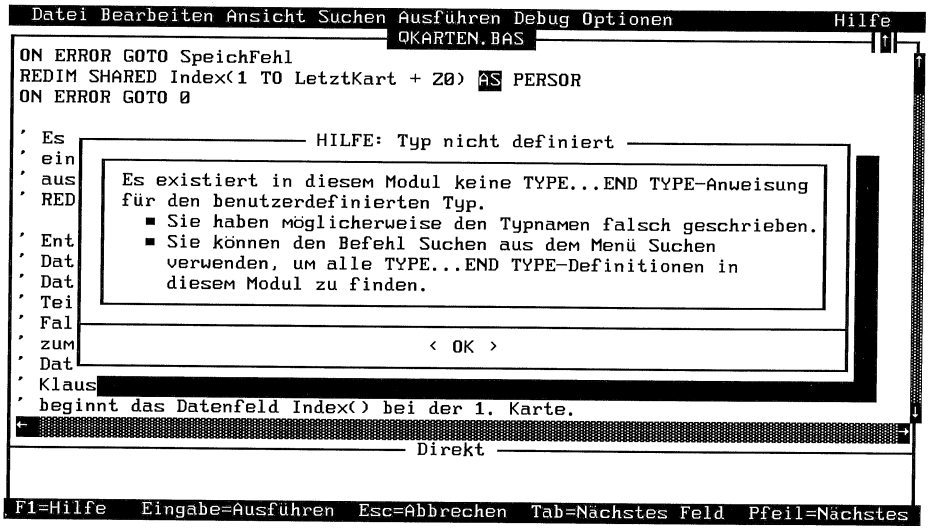
Hilfe für Fehlermeldungen

Die Dialogfelder für Fehlermeldungen weisen eine Hilfsfläche auf, die bei Auswahl entweder eine Beschreibung der häufigsten Fehlerursachen oder die korrekte Syntax und Anwendungsbeispiele anzeigt.

1. Fordern Sie die On-Line-Hilfe für diese Fehlermeldung durch Betätigen der Taste F1 an, und wählen Sie dann die Hilfsfläche des Dialogfeldes an.

Es ist jetzt folgendes Dialogfeld zu sehen (siehe Abbildung 6.3):

Abbildung 6.3 Hilfe für Fehlermeldungen



Das neue Dialogfeld gibt an, daß keine **TYPE...END TYPE**-Anweisung für `PERSOR` gefunden wurde.

2. Drücken Sie die ESC-TASTE so oft wie erforderlich, um das Dialogfeld zu löschen.

Überschreiben des Fehlers

Durch Überschreiben wird der Text an der Cursorstelle ersetzt und der Cursor nach rechts bewegt. Der Fehler ist durch Überschreiben wie folgt zu verbessern:

6. Code-Eingabe in QKARTEN

1. Positionieren Sie den Cursor auf dem zweiten `R` in `PERSOR`.

2. Gehen Sie vom Einfüge- in den Überschreibmodus durch Betätigen der EINFG-TASTE über.

   Die EINFG-TASTE stellt einen "Umschalter" dar und ändert die Standardform des Cursors (eine dünne blinkende Linie) in ein blinkendes Feld um. Wenn der Cursor die Form einer dünnen Linie annimmt, wird der eingegebene Text an der Cursorstelle eingefügt. Nimmt er dagegen die Form eines Feldes an, "überschreibt" (ersetzt) die Eingabe den an der aktuellen Cursorstelle vorhandenen Text.

3. Überschreiben Sie das zweite `R` in `PERSOR` mit einem `N`, so daß die Zeile wie folgt aussieht:

   ```
   REDIM SHARED Index(1 TO LetztKart + 20) AS PERSON
   ```

4. Schalten Sie durch Betätigen der EINFG-TASTE wieder auf den Einfügemodus um (dünne Linie).

5. Rufen Sie die Prozedur `Alarm` durch folgende Eingabe in der nächsten Leerzeile auf:

   ```
   call Alarm
   ```

6. Drücken Sie ALT+A und S, um den Befehl **Start** aus dem Menü **Ausführen** auszuwählen und die Zeilen auszuführen.

   QKARTEN erzeugt zwar noch keine sichtbare Ausgabe, aber `Alarm` gibt bereits ein Signal aus.

7. Kehren Sie durch Betätigen einer Taste in die QuickBASIC-Umgebung zurück.

Automatische Prozedur-Vereinbarungen

Beim Antreffen eines Prozeduraufrufs in einem Programm erstellt QuickBASIC automatisch eine **DECLARE**-Anweisung. Anschließend schreibt es beim Speichern des Programms die Vereinbarung wie folgt in die erste Zeile der Datei:

7. Code-Eingabe in QKARTEN

1. Gehen Sie durch Betätigen der Tastenkombination STRG+POS1 an den Anfang des Programms.
2. Drücken Sie ALT+D und U, um den Befehl **Speichern unter...** aus dem Menü **Datei** auszuwählen und das Dialogfeld **Speichern unter** erscheinen zu lassen.
3. Geben Sie `Kapitel6` im Textfeld ein, und speichern Sie das Programm durch Betätigen der EINGABETASTE unter dem Namen *kapitel6.bas*. QuickBASIC fügt die Vereinbarung wie folgt in der ersten Programmzeile ein:

   ```
   DECLARE SUB Alarm ()
   ```

Die Arbeit mit ausgewählten Textblöcken

Das ist der richtige Zeitpunkt für eine Pause. Im nächsten Abschnitt werden Sie die neu erstellte Vereinbarung verschieben und Teile des bestehenden QKARTEN-Codes ändern.

"Auswählen" bezeichnet das Markieren eines zu bearbeitenden Textblocks. Um einen Textblock direkt auszuwählen, positionieren Sie den Cursor an den Anfang eines Blocks, und betätigen Sie die geeignete RICHTUNGSTASTE bei niedergedrückter UMSCHALTTASTE.

Ein Textblock läßt sich wie folgt auswählen:

8. Code-Eingabe in QKARTEN

1. Starten Sie QuickBASIC falls erforderlich, und laden Sie *kapitel6.bas*.
2. Vergewissern Sie sich durch Betätigen der Tastenkombination STRG+POS1, daß sich der Cursor in der ersten Programmzeile befindet.

   Es müßte die Zeile `DECLARE SUB Alarm ()` sein.
3. Wählen Sie diese Zeile mit Hilfe der Tastenkombination UMSCHALT+NACH UNTEN aus.

Die einzige Möglichkeit, die inverse Darstellung des ausgewählten Texts zu löschen und dessen ursprünglichen Zustand wiederherzustellen besteht im Betätigen einer RICHTUNGSTASTE (NACH OBEN, NACH UNTEN, NACH LINKS, NACH RECHTS) oder der Taste F6, um sich zu einem anderen Fenster zu bewegen.

Warnung Mit ausgewähltem Text ist Vorsicht geboten! Während eine Textzeile ausgewählt ist, ersetzt jegliche Eingabe (sogar das Betätigen der LEERTASTE) den gesamten im aktiven Fenster ausgewählten Text, der damit endgültig verloren geht.

Ausschneiden und Kopieren des ausgewählten Texts

Das Menü **Bearbeiten** verfügt über Befehle, die das Einfügen eines Textblocks in einen und das Kopieren aus einem "Text-Zwischenspeicher" ermöglichen. Die Befehle **Ausschneiden** und **Kopieren** kopieren ausgewählten Text in den Zwischenspeicher. Der Befehl **Einfügen** kopiert den im Zwischenspeicher vorhandenen Text an die aktuelle Cursorstelle. Wenn eine oder mehrere Textzeilen bei Auswahl des Befehls **Einfügen** im aktiven Fenster markiert sind, ersetzt der Text im Zwischenspeicher den markierten Text.

Ein verfügbarer Menübefehl ist durch intensiv dargestellte Buchstaben gekennzeichnet. Wird beispielsweise das Menü **Bearbeiten** zu diesem Zeitpunkt geöffnet, sind die Befehle **Ausschneiden** und **Kopieren** verfügbar, da im aktiven Fenster ausgewählter Text vorhanden ist. Der Befehl **Einfügen**, dagegen, ist nicht verfügbar, solange kein Text in den Zwischenspeicher kopiert wurde. Nachstehende Arbeitsschritte übertragen den Text mit Hilfe des Befehls **Ausschneiden** von der aktuellen Position in den Zwischenspeicher:

9. Code-Eingabe in QKARTEN

1. Wählen Sie die Zeile `DECLARE SUB Alarm` aus, falls diese nicht bereits ausgewählt ist.
2. Öffnen Sie das Menü **Bearbeiten** falls erforderlich durch Betätigen der Tastenkombination ALT+B.
3. Wählen Sie den Befehl **Ausschneiden** durch Drücken der Taste A aus.

Der ausgewählte Text wird in den Zwischenspeicher übertragen. In den Zwischenspeicher kann jeweils nur ein Textblock kopiert werden, da jeder neue Block unabhängig von seiner Größe den zuvor im Zwischenspeicher vorhandenen Block ersetzt.

Wichtig Die Tastenkombination zur direkten Befehlseingabe von **Ausschneiden** ist UMSCHALT+ENTF. Ausgewählter Text wird durch Betätigen der ENTF-TASTE gelöscht, ohne in den Zwischenspeicher kopiert zu werden. Über die ENTF-TASTE gelöschter Text kann nicht wiederhergestellt werden.

Der Befehl **Kopieren** ist dem Befehl **Ausschneiden** ähnlich, mit dem Unterschied, daß dieser ein Duplikat des ausgewählten Texts in den Zwischenspeicher überträgt, wobei der Originalblock an der alten Stelle bleibt.

Hinweis Der Befehl **Kopieren** beläßt den Originalblock markiert, so daß vor jeglicher weiteren Eingabe eine RICHTUNGSTASTE zu betätigen ist. Andernfalls wird eine neue Eingabe den Block ersetzen.

Einfügen eines Textblocks

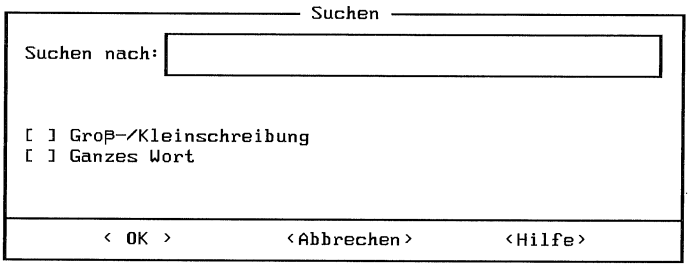
Jeder in den Zwischenspeicher kopierte Text wird dort beibehalten, bis er durch einen anderen Text ersetzt oder QuickBASIC beendet wird. Gehen Sie mit Hilfe nachstehender Arbeitsschritte zu dem QKARTEN-Abschnitt mit Prozedur-Vereinbarungen, und fügen Sie die Vereinbarung für `Alarm` ein.

10. Code-Eingabe in QKARTEN

1. Drücken Sie ALT+S und S, um den Befehl **Suchen...** aus dem Menü **Suchen** auszuwählen.

   Es erscheint das Dialogfeld **Suchen** (siehe Abbildung 6.4)

Abbildung 6.4 Das Dialogfeld Suchen



2. Geben Sie folgende Zeile im Textfeld ein:

   ```
   Dieser Platz ist reserviert
   ```

3. Beginnen Sie die Suche durch Betätigen der EINGABETASTE.
4. Positionieren Sie den Cursor durch Drücken der POS1-TASTE in der ersten Spalte.
5. Wählen Sie die gesamte Zeile mit Hilfe der Tastenkombination UMSCHALT+ENDE aus.
6. Drücken Sie ALT+B und E, um den Befehl **Einfügen** aus dem Menü **Bearbeiten** auszuwählen.

   Der Befehl **Einfügen** ist jetzt aktiv, da sich Text im Zwischenspeicher befindet. Der Text aus dem Zwischenspeicher ersetzt den markierten Text.
7. Speichern Sie das Programm unter dem Namen *kapitel6.bas*.

Manipulieren des ausgewählten Texts

Es gibt auch andere Möglichkeiten der Textmanipulation. Ein ausgewählter Text läßt sich zum Beispiel block- anstatt zeilenweise einrücken, wie aus folgendem Abschnitt ersichtlich.

11. Code-Eingabe in QKARTEN

1. Drücken Sie ALT+S und S, um den Befehl **Suchen...** aus dem Menü **Suchen** auszuwählen. Geben Sie folgende Zeile genau wie angegeben im Textfeld ein:

   ```
   ' Definiere zeitweilig
   ```

   Beim Öffnen des Dialogfeldes kann möglicherweise im Textfeld eine ausgewählte Textpassage vorhanden sein. In diesem Fall würde eine neue Eingabe diesen Text automatisch ersetzen. Falls er nicht vollständig ersetzt werden soll, können Sie durch Betätigen der RICHTUNGSTASTEN den im Feld vorhandenen Text lediglich ändern.

2. Bringen Sie den Cursor durch Drücken der EINGABETASTE in die folgende Zeile:

   ```
   ' Definiere zeitweilig das Datenfeld Index(), um die
   ' QKARTEN-Anzeige zu veranschaulichen
   ```

3. Wählen Sie folgende Zeilen durch dreimaliges Betätigen der Tastenkombination UMSCHALT+NACH UNTEN aus:

   ```
   ' Definiere zeitweilig das Datenfeld Index(), um die
   ' QKARTEN-Anzeige zu veranschaulichen
   REDIM SHARED Index(1) AS PERSON
   ```

4. Drücken Sie die TAB-TASTE, während der Textblock ausgewählt bleibt. Der gesamte Block bewegt sich um einen Tabulatorstopp nach rechts.

5. Bewegen Sie den Block durch Betätigen der Tastenkombination UMSCHALT+TAB um eine Einzugsebene nach links.

6. Löschen Sie den Block endgültig anhand der ENTF-TASTE.

7. Speichern Sie das Programm wie üblich unter dem Namen *kapitel6.bas*.

Es ist zu beachten, daß der gelöschte Text nicht in den Zwischenspeicher kopiert wurde, da er anhand der ENTF-TASTE, nicht der Tastenkombination UMSCHALT+ENTF für **Ausschneiden** gelöscht wurde. Die temporäre Definition des Datenfeldes `Index()` wird nicht mehr benötigt, da dieses im ersten Teil dieses Kapitels mit einer `REDIM`-Anweisung definiert wurde.

Suchen und Ersetzen von Text

Wichtig Die restlichen Abschnitte dieses Kapitels behandeln die QuickBASIC-Einrichtungen **Suchen** und **Ersetzen**. Die nachstehenden optionalen Codeeingaben sind zur Ausführung des Programms QKARTEN nicht erforderlich. Es ist empfehlenswert, den Rest des Kapitels vor Eingabe des Codes zu lesen. Ein Fehler in der Eingabe kann die korrekte Ausführung des endgültigen Programms beeinträchtigen. Wenn Sie sich zur Eingabe entscheiden, müssen Sie alle drei Code-Eingaben vornehmen.

In diesem Abschnitt werden Sie die symbolische Konstante `DATEI$` zur Darstellung von "*qkarten.dat*", dem Namen der Datei mit den QKARTEN-Daten, definieren. Anschließend werden Sie anhand des Befehls **Ändern** aus dem Menü **Suchen** alle Vorkommen der Textzeichenkette "*qkarten.dat*" mit der symbolischen Konstante ersetzen. Nach Definieren der Konstante `DATEI$`, können Sie die Definition von `DATEI$` verändern, wenn Sie eine andere Daten-Datei in QKARTEN laden oder eine neue Daten-Datei anlegen möchten.

Definieren einer symbolischen Konstante

Das erste zu definierende Element ist die Konstante. Im Vereinbarungs- und Definitionsabschnitt gibt es bereits einen für diesen Vorgang geeigneten Platz, der mit Hilfe folgender Arbeitsschritte gesucht und mit der Konstanten ergänzt werden kann:

1. optionale Code-Eingabe in QKARTEN

1. Speichern Sie *kapitel6.bas* unter dem Namen *optional.bas*.

   Auf diese Weise können Sie das nächste Kapitel mit dem Laden von *optional.bas* und anschließendem Speichern unter dem Namen *kapitel7.bas* beginnen. Falls Sie jedoch einen Fehler in den optionalen Abschnitten machen und diesen nicht verbessern können, haben Sie die Möglichkeit, das nächste Kapitel mit *kapitel6.bas* zu beginnen.
2. Gehen Sie an den Anfang von QKARTEN durch Betätigen der Tastenkombination STRG+POS1.
3. Drücken Sie ALT+S und S, um den Befehl **Suchen...** aus dem Menü **Suchen** auszuwählen, und geben Sie folgenden Text im Textfeld ein:

   ```
   TempDat
   ```
4. Positionieren Sie den Cursor durch Drücken der EINGABETASTE an der Definition von `TMPFILE`, die wie folgt aussehen sollte:

   ```
   CONST TempDat = "$$$87y$.$5$"       ' Unwahrscheinlicher
                                       ' Dateiname
   ```
5. Entfernen Sie die Markierung von TempDat durch Betätigen der NACH UNTEN TASTE.
6. Drücken Sie die POS1-TASTE, um den Cursor in der äußersten linken Spalte zu plazieren, und geben Sie folgende Zeile genau wie angegeben ein, ohne die EINGABETASTE zu betätigen:

   ```
   CONST DATEI$ = "QKARTEN.DAT"
   ```
7. Betätigen Sie UMSCHALT+NACH LINKS, bis die Zeichenkette `"QKARTEN.DAT"` vollständig markiert ist. Vergewissern Sie sich, daß die Markierung beide Anführungszeichen einschließt.

Es ist wichtig, die Markierung der Anführungszeichen sicherzustellen, da `"QKARTEN.DAT"` durch `DATEI$` ersetzt werden soll.

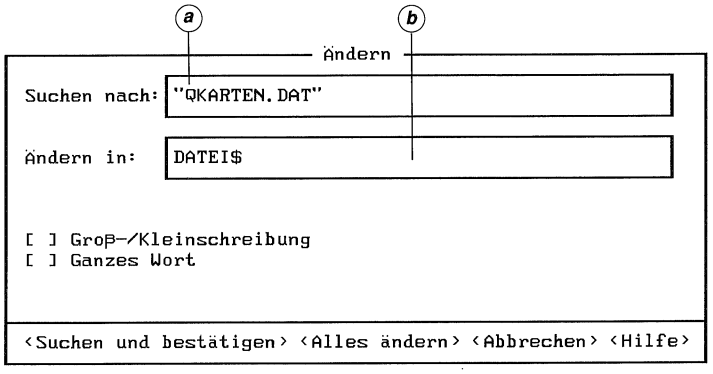
Ersetzen mehrerer Textvorkommen

Der Befehl **Ändern** ist dem Befehl **Suchen** sehr ähnlich, mit dem Unterschied, daß QuickBASIC das Ersetzen des angegebenen Texts beim Antreffen eines Suchbegriffs ermöglicht. Als Nächstes werden Sie "QKARTEN.BAS" in DATEI$ überall mit Ausnahme der im letzten Abschnitt geschriebenen Definition umändern. Der Befehl **Ändern...** aus dem Menü **Suchen** automatisiert diesen Vorgang.

2. optionale Code-Eingabe in QKARTEN

1. Drücken Sie ALT+S und D, um den Befehl **Ändern** aus dem Menü **Suchen** auszuwählen. Es erscheint das Dialogfeld **Ändern** (siehe Abbildung 6.5):

Abbildung 6.5 Das Dialogfeld Ändern



a) Der zu ändernde Text wird an dieser Stelle eingegeben.
b) Dieser Text wird den zu ändernden Text ersetzen.

Da "QKARTEN.DAT" im Arbeitsfenster ausgewählt ist, enthält das Textfeld **Suchen nach** den Namen "QKARTEN.DAT".

2. Bringen Sie den Cursor mit Hilfe der TAB-TASTE in das Textfeld **Ändern in**.

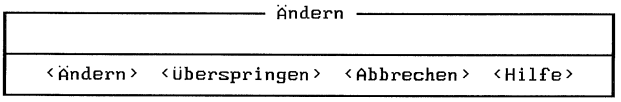
   An dieser Stelle wird der Ersatztext für den gesuchten Text eingegeben.

3. Geben Sie DATEI$ in diesem Textfeld ein.
4. Wählen Sie die Standard-Befehlsfläche **Suchen und bestätigen** mit Hilfe der EINGABETASTE aus.

Warnung Die zweite Befehlsfläche, **Alles ändern**, nimmt alle Änderungen vor, ohne die Vorkommen einzublenden oder eine Bestätigung abzuwarten. Diese Option ist risikoreich, wenn Sie sich nicht vergewissern, daß sämtliche Vorkommen geändert werden sollen. In diesem Fall kann es vorkommen, daß Sie versehentlich Wörter, die den gesuchten Text enthalten, in einer unvorsehbaren Weise verändern. Im Zweifelsfall ist die Option **Suchen und bestätigen** zu verwenden. Die Befehlsfläche **Abbrechen** oder die ESC-TASTE brechen das Suchen und Ersetzen völlig ab.

Anhand der Befehlsfläche **Suchen und bestätigen** werden sämtliche Vorkommen des gesuchten Texts hervorgehoben und das Dialogfeld **Ändern, Überspringen, Abbrechen** angezeigt (siehe Abbildung 6.6). Im nächsten Abschnitt werden Sie mit Hilfe dieses Dialogfeldes Textpassagen selektiv ersetzen. Es verfügt über Befehlsflächen, die nach der Entscheidung bezüglich der zu ersetzenden Textpassage ausgewählt werden können.

Abbildung 6.6 Das Dialogfeld Ändern, Überspringen, Abbrechen



Das Dialogfeld **Ändern, Überspringen, Abbrechen** ist weiterhin auf dem Bildschirm zu sehen. Die Vorkommen von "QKARTEN.DAT" sind wie folgt durch DATEI$ zu ersetzen:

3. optionale Code-Eingabe in QKARTEN

1. Drücken Sie die Taste N, um das erste Vorkommen zu ersetzen, das sich in der vorher in diesem Kapitel eingegebenen OPEN-Anweisung befindet.

   QuickBASIC springt zum nächsten Vorkommen des Suchbegriffs, das sich in einer Kommentarzeile der Prozedur Aktualisieren befindet (überprüfen Sie die Titelleiste des Arbeitsfensters).

2. Nehmen Sie die Änderung durch Betätigen der Taste N vor.

   QuickBASIC springt zum nächsten Vorkommen des Suchbegriffs, das sich ebenfalls in der Prozedur `Aktualisieren` in der Zeile `KILL "QKARTEN.DAT"` befindet.

3. Nehmen Sie die Änderung durch Betätigen der Taste N vor.

   QuickBASIC springt zum nächsten Vorkommen des Suchbegriffs, das sich ebenfalls in der Prozedur `Aktualisieren` in der Zeile `NAME TMPFILE AS "QKARTEN.DAT"` befindet.

4. Nehmen Sie die Änderung durch Betätigen der Taste N vor.

   QuickBASIC springt zum nächsten Vorkommen des Suchbegriffs, das sich in der Kommentarzeile einer anderen Prozedur, `InitIndex`, befindet.

5. Nehmen Sie die Änderung durch Betätigen der Taste N vor.

   Das letzte Vorkommen befindet sich am Anfang, in `CONST DATEI$ = "QKARTEN.DAT"`. Dieses Vorkommen soll jedoch *nicht* geändert werden, da alle anderen Änderungen wegen dieser Definition vorgenommen wurden.

6. Wählen Sie die Befehlsfläche **Überspringen** durch Betätigen der Taste S aus.

   Nachdem sämtliche Vorkommen entweder geändert oder übersprungen wurden, gibt QuickBASIC ein Dialogfeld mit dem Text `Änderung ausgeführt` aus.

7. Schließen Sie das Dialogfeld über die ESC-TASTE.

Überprüfen der Arbeit

Nach Durchführung einer Reihe von Änderungen im gesamten Programm sollten Sie Ihre Arbeit durch einen Programmablauf überprüfen. Falls Ihnen während der Bearbeitung ein Fehler unterlaufen ist, können Sie diesen mit Hilfe der QuickBASIC-Fehlermeldungen verfolgen. Schließen Sie diese Arbeitssitzung mit folgendem Verfahren ab:

1. Drücken Sie ALT+A und S, um den Befehl **Start** aus dem Menü **Ausführen** auszuwählen und die Zeilen auszuführen.

   Beim Auftreten einer Fehlermeldung versuchen Sie, den Fehler zu entdecken und vor Beenden der Arbeitssitzung zu verbessern. Gehen Sie jede optionale Code-Eingabe in QKARTEN erneut durch, um Ihre Arbeit damit zu vergleichen.

2. Kehren Sie durch Betätigen einer Taste in die QuickBASIC-Umgebung zurück.

3. Speichern Sie das Programm. Wenn Sie es in *optional.bas* umbenannt haben, können Sie das nächste Kapitel mit dieser Datei beginnen und sie *kapitel7.bas* nennen.

Die Konstante `DATEI$` stellt jetzt *qkarten.dat* im Programm dar. Später empfiehlt es sich, das Programm eventuell zu ändern, so daß der Benutzer während des Programmablaufs mehrere Dateien laden kann. In diesem Fall darf die Datei nicht durch die Konstante DATEI$ dargestellt werden, da eine Konstante während des Programmablaufs nicht verändert werden kann. Aus diesem Grund werden Sie den `CONST`-Teil der Definition entfernen und `DATEI$` in eine normale Zeichenkettenvariable umändern müssen.

Weitere Informationen

Weitere Informationen über die in diesem Kapitel behandelten Themen sind folgenden Kapiteln zu entnehmen:

Kapitel	***Themen***
Kapitel 12, "Die Verwendung des Editors", und Kapitel 13, "Das Menü Bearbeiten"	Vollständige Informationen über die QuickBASIC-Editiereinrichtungen, einschließlich der Befehle in den vollständigen Menüs und der Tastenkombinationen zur direkten Befehlseingabe.
Kapitel 2, "Prozeduren: Unterprogramme und Funktionen", aus dem Handbuch *Programmieren in BASIC*	Definieren und Verwenden von Prozeduren und Einsatz der automatisch erzeugten Prozedurvereinbarungen.
Kapitel 4, "Zeichenketten-verarbeitung", aus dem Handbuch *Programmieren in BASIC*	Verwenden von Zeichenketten als symbolische Konstanten.
Kapitel 6, "Fehler- und Ereignisverfolgung", aus *Programmieren in BASIC*	Verwenden der **ON ERROR**-Anweisung.

7 Programmieren mit der On-Line-Hilfe

Die Eigenschaften der On-Line-Hilfe von Microsoft QuickBASIC haben Sie bereits in den vorhergehenden Kapiteln kennengelernt. In diesem Kapitel werden Sie die On-Line-Hilfe als Anleitung zum Schreiben von Code für das Programm QKARTEN wie folgt benutzen:

- Verwendung alternativer Methoden zum Anfordern der On-Line-Hilfe für QuickBASIC-Schlüsselwörter.
- Zugriff auf Informationen zu den On-Line-Programmsymbolen anhand der On-Line-Hilfe.
- Anfordern von zusätzlichen Informationen innerhalb des Hilfsfensters anhand der Hyperverbindungen.

Dieses Kapitel erfordert etwa eine Arbeitsstunde.

Aufbau von Anweisungen anhand der On-Line-Hilfe

In diesem Abschnitt wird der Einsatz von Anweisungen anhand des Befehls **Index** aus dem Menü **Hilfe** verdeutlicht. Verfahren Sie zunächst wie folgt:

12. Code-Eingabe in QKARTEN

1. Starten Sie QuickBASIC falls erforderlich, und laden Sie *kapitel6.bas* (oder *optional.bas*).
2. Speichern Sie das Programm unter dem Namen *kapitel7.bas*.

On-line-Hilfe für Schlüsselwörter

Eine Anweisung, die Sie im Programm benutzen wollen, können Sie sehr schnell durch Auswahl des Befehls **Index** aus dem Menü **Hilfe** finden. Durch nachstehende Arbeitsschritte können Sie Informationen über den ersten in diesem Kapitel einzugebenden Code-Block (die Anweisung **IF...THEN...ELSE**) erhalten:

1. Wählen Sie den Befehl **Index** aus dem Menü **Hilfe** durch Drücken der Tastenkombination ALT+H und I aus.

   Index führt alle QuickBASIC-Schlüsselwörter auf. Sehen Sie sich die Liste mit Hilfe der RICHTUNGSTASTEN an, oder drücken Sie den Anfangsbuchstaben des Schlüsselwortes, zu dem Sie weitere Informationen benötigen.

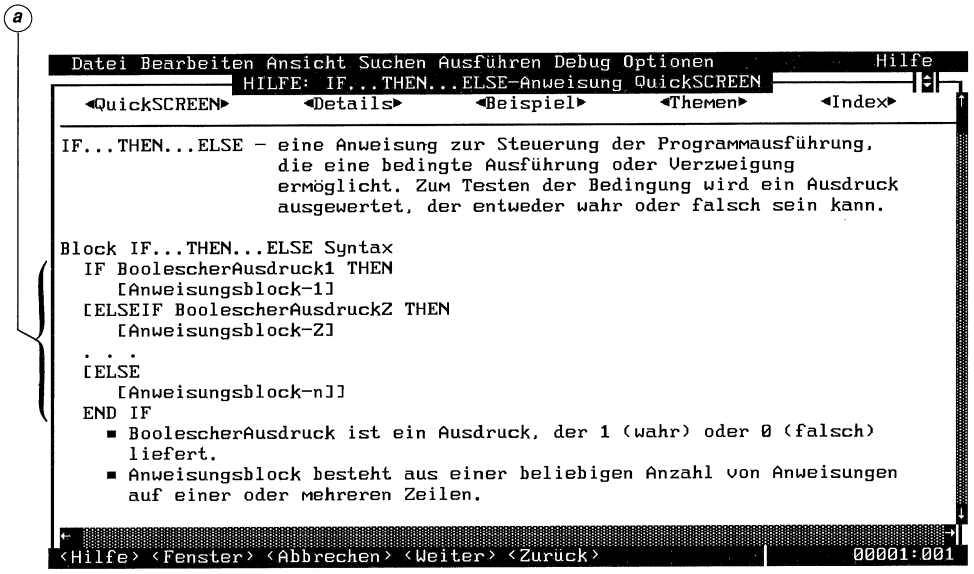
2. Wählen Sie `IF...THEN` durch Betätigen der Taste I aus.

   QuickBASIC plaziert den Cursor in der Anweisung `IF...THEN`.

3. Fordern Sie Hilfe für `IF...THEN` durch Betätigen der Taste F1 an.

   Auf dem Bildschirm (siehe Abbildung 7.1) erscheint ein Hilfebildschirm, der eine Kurzbeschreibung der Anweisung, die Syntax der blockförmigen **IF...THEN...ELSE** und eine Beschreibung der in diesem Block verwendeten Begriffe enthält.

Abbildung 7.1 Hilfebildschirm für die Anweisung IF...THEN...ELSE



a) *Syntaxblock.*

4. Erweitern Sie das Hilfsfenster durch Betätigen der Tastenkombination STRG+F10 auf die Gesamtbildschirmgröße.
5. Gehen Sie durch Drücken der BILD NACH UNTEN TASTE bis zur Syntax der einzeiligen `IF...THEN...ELSE`.
6. Bringen Sie den Cursor durch Betätigen der Tastenkombination STRG+POS1 zum oberen Ende des Hilfebildschirms zurück.
7. Stellen Sie das Arbeits- und das Direktfenster durch Betätigen der Tastenkombination STRG+F10 wieder her.

Wenn Ihnen eine Anweisung bereits bekannt ist, genügt die Information des Hilfebildschirms zum Schreiben von Code.

Hyperverbindungen in der On-Line-Hilfe

Der Hilfebildschirm weist vier Hyperverbindungen am oberen Ende auf. Sie können den Cursor auf eine dieser Verbindungen bringen, indem Sie die TAB-TASTE oder den Anfangsbuchstaben des Wortes in der Hyperverbindung drücken. Nachstehende Liste beschreibt die Anzeigen sämtlicher Hyperverbindungen:

Hyperverbindung	*Von Hilfe angezeigte Information*
Details	Erläutert ausführlich jedes Syntaxelement und beschreibt alle Einsatzmöglichkeiten einer Anweisung
Beispiel	Zeigt Code-Beispiele zur Veranschaulichung einer Syntaxanwendung
Themen	Zeigt alle in QuickBASIC verfügbaren Kategorien der On-Line-Hilfe an
Index	Listet QuickBASIC-Schlüsselwörter in alphabetischer Reihenfolge auf

Sehen Sie sich alle verfügbaren Hyperverbindungen an, und drücken Sie ALT+F1, um zu der Syntaxanzeige zurückzukehren.

Wenn Sie dagegen mehrere Hyperverbindungen anwählen, ohne zur vorhergehenden Anzeige zurückzukehren, können Sie durch wiederholtes Betätigen der Tastenkombination ALT+F1 die Folge der Hyperverbindungen (bis zu 20 Verbindungen) zurückverfolgen.

Obwohl das Hilfsfenster keine Eingabe zuläßt, kann dessen Größe anhand der Tastenkombination ALT+PLUS und ALT+MINUS verändert werden, um die benötigte Information während der Texteingabe im Arbeitsfenster anzuzeigen. Nachdem Sie die blockförmige **IF...THEN...ELSE**-Anweisung verstanden haben, können Sie den im Kommentarblock beschriebenen Programmierabschnitt wie folgt implementieren.

13. Code-Eingabe in QKARTEN

1. Bringen Sie den Cursor durch Betätigen der Taste F6 in das Arbeitsfenster.
2. Gehen Sie zu der Zeile unterhalb des nachstehenden Kommentarblocks (ungefähr Zeile 145), und erstellen Sie eine Leerzeile mit der EINGABETASTE:

```
' Entscheide mit Hilfe der blockförmigen
' IF...THEN...ELSE-Anweisung, ob die Datensätze aus der
' Diskettendatei QKARTEN in das vorher deklarierte
' Datensatzfeld mit dem Namen Index() geladen werden
' sollen. Prüfe im IF-Teil, ob tatsächlich Datensätze in
' der Datei vorhanden sind. In diesem Fall wird LetztKart
' größer als Null sein, und die Prozedur InitIndex kann
' zum Laden der Datensätze in Index() aufgerufen werden.
' Wenn es noch keine Datensätze in der Datei gibt, nimmt
' LetztKart den Wert 0 an, und die ELSE-Klausel wird
' ausgeführt. Der zwischen ELSE und END IF vorhandene
' Code beginnt das Datenfeld Index() bei der 1. Karte.
```

3. Geben Sie folgenden Code genau wie angegeben ein:

```
if LetztKart <> 0 then
    call InitIndex(LetztKart)
else
    Karte.KartNum = 1
    Index(1) = Karte
    put #1, 1, Karte
end if
```

4. Speichern Sie das Programm unter dem Namen *kapitel7.bas*, und bringen Sie den Cursor auf das Schlüsselwort `ELSE` zurück.
5. Führen Sie das Programm durch Betätigen der Taste F7 bis zu der Zeile aus, in der sich der Cursor befindet.

   Diese Zeile wird jetzt in den Farben hervorgehoben, die für die Option **Aktuelle Anweisung** im Dialogfeld **Bildschirm** aus dem Menü **Optionen** ausgewählt wurden. Das bedeutet, daß das Programm weiterhin aktiv ist, aber die Ausführung zeitweilig unterbrochen wurde.

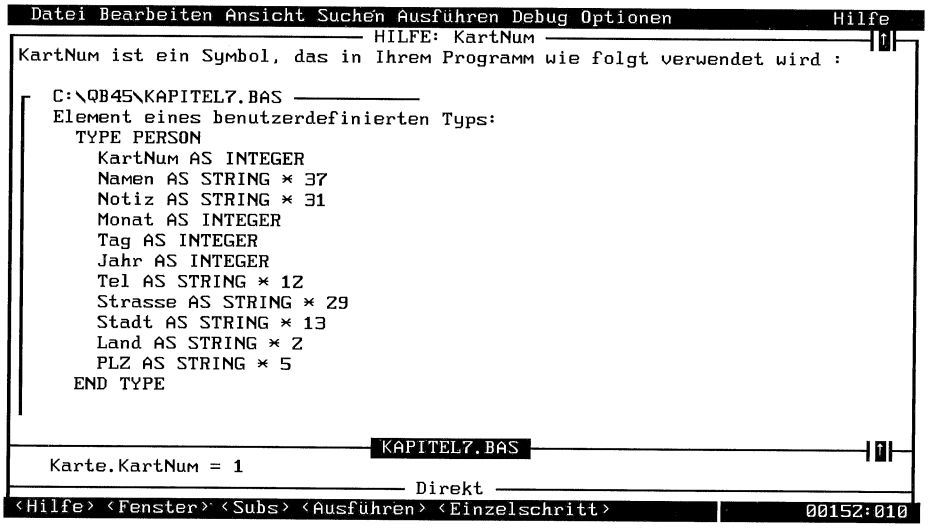
On-Line-Hilfe für Programmsymbole

QuickBASIC erteilt Hilfe für sämtliche "Symbole" des Programms. Ein Symbol stellt jeglichen explizit oder implizit definierten Begriff dar. QuickBASIC-Schlüsselwörter sind beispielsweise von der Programmiersprache definierte Symbole. Um Hilfe für die im Programm definierten Symbole anzufordern, ist der Cursor innerhalb des Symbols zu positionieren und die Taste F1 zu betätigen. Anschließend gibt QuickBASIC vollständige Informationen über das Symbol und dessen Einsatz im Programm aus. Üben Sie dieses Verfahren, indem Sie Informationen zu `Karte.KartNum` anfordern.

1. Positionieren Sie den Cursor innerhalb des Wortes `KartNum` in `Karte.KartNum`, und drücken Sie F1.

   Es erscheint das in Abbildung 7.2 dargestellte Hilfsfenster.

Abbildung 7.2 Symbolhilfe für KartNum



Die On-Line-Hilfe beschreibt `KartNum` als ein Element des benutzerdefinierten Typs `PERSON` und zeigt die Anweisung **TYPE...END TYPE** an, deren erstes Element `KartNum` ist.

2. Bewegen Sie den Cursor nach links, bis er sich im Wort `Karte` auf der linken Seite des Punktes befindet, und drücken Sie F1. Die On-Line-Hilfe zeigt, daß die Bezeichnung `Karte` in QKARTEN häufig verwendet wird. Zuerst wird das Vorkommen der Bezeichnung an der Cursorstelle während der Betätigung der Taste F1 als eine Variable des Typs `PERSON` beschrieben.

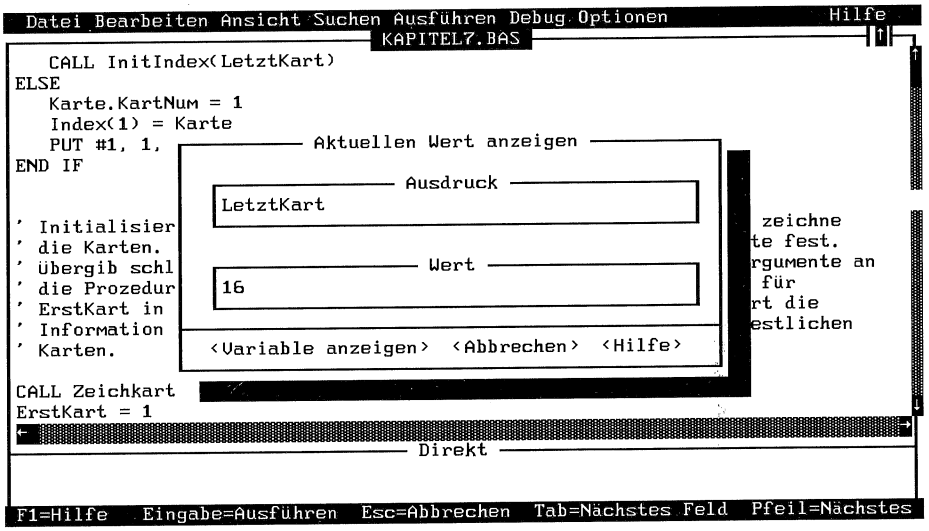
   Es ist zu beachten, daß `Karte` in einigen Prozeduren als Variablenname des Typs `INTEGER` verwendet wird. `Karte` wird in einigen Fällen auch als ein Parameter des Typs `PERSON` beschrieben.

Die On-Line-Hilfe erleichtert erheblich die Verfolgung der im Programm verwendeten Variablennamen. Obwohl die On-Line-Hilfe den aktuellen Wert einer Variablen nicht anzeigt, vereinfacht QuickBASIC die Bestimmung des aktuellen Wertes einer Variablen in einem unterbrochenen Programm auf folgende Weise:

1. Schließen Sie das Hilfsfenster durch Betätigen der ESC-TASTE.
2. Positionieren Sie den Cursor innerhalb des Wortes `LetztKart` (ungefähr Zeile 147).
3. Wählen Sie den Befehl **Aktuellen Wert anzeigen** aus dem Menü **Debug**.

   QuickBASIC zeigt das Dialogfeld **Aktuellen Wert anzeigen** (siehe Abbildung 7.3) an, das den Variablennamen `LetztKart` und deren aktuellen Wert enthält. (Der aktuelle Wert entspricht der Datensatzanzahl in der Daten-Datei *qkarten.bas*.)

Abbildung 7.3 Das Dialogfeld Aktuellen Wert anzeigen



4. Löschen Sie das Dialogfeld **Aktuellen Wert anzeigen** durch Betätigen der ESC-TASTE.

Lassen Sie mit Hilfe folgender Arbeitsschritte die Karten des Programms QKARTEN auf dem Bildschirm und die Daten aus *qkarten.bas* in den Karten erscheinen:

14. Code-Eingabe in QKARTEN

1. Positionieren Sie den Cursor in der Zeile unterhalb des folgenden Kommentars (ungefähr Zeile 163):

```
' Initialisiere die Anzeige mit Hilfe der Prozedur
' ZeichKart und zeichne die Karten. Setze anschließend
' die erste Karte als oberste Karte fest. Übergib
' schließlich die Variablen ErstKart und LetztKart als
' Argumente an die Prozedur ZeigKart. Der Aufruf von
' ZeigKart setzt alle Daten für ErstKart in die erste
' Karte auf dem Bildschirm ein und die Information der
' obersten Zeile (der Name der Person) in die restlichen
' Karten.
```

2. Erstellen Sie eine Leerzeile über die EINGABETASTE, und geben Sie nachstehenden Code genau wie angegeben ein:

```
call ZeichKart
ErstKart = 1
call ZeigKart(ErstKart, LetztKart)
```

3. Bringen Sie den Cursor in die letzte eingegebene Zeile (die mit `CALL ZeigKart` beginnt) zurück.
4. Führen Sie das Programm durch Betätigen der Taste F7 bis zu der Zeile aus, in der sich der Cursor befindet.
5. Wählen Sie den Befehl **Ausgabebildschirm** aus dem Menü **Ansicht** aus, um zwischen der QuickBASIC-Umgebung und dem QKARTEN-Ausgabebildschirm durch Betätigen der Taste F4 zu wechseln.

Sie können die oben erwähnte Technik zum Anfordern von Hilfe für das Schlüsselwort **CALL** verwenden, das die Kontrolle an eine BASIC-**SUB**-Prozedur übergibt. Sie können ebenfalls Hilfe für die Prozeduren `ZeichKart` und `ZeigKart` und für die Variable `ErstKart` erhalten. Da die Ausführung des Programms nur unterbrochen ist, können die Werte seiner Variablen mit Hilfe des Befehls **Aktuellen Wert anzeigen** aus dem Menü **Debug** überprüft werden.

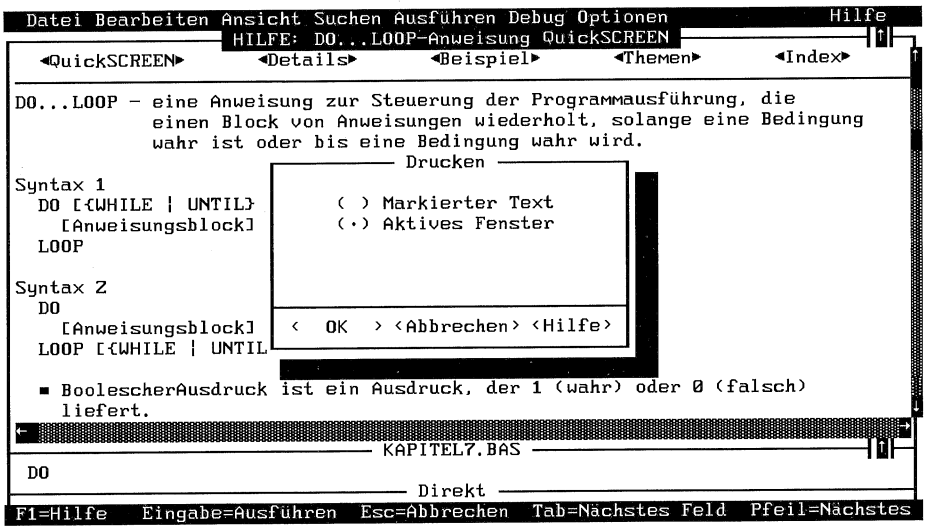
Drucken von Anzeigen aus der On-Line-Hilfe

Anhand des Befehls **Drucken...** aus dem Menü **Datei** können gedruckte Textausgaben der On-Line-Hilfe erzeugt werden. Wenn am parallelen Anschluß ein Drucker angeschlossen ist, lassen sich gedruckte Ausgaben sämtlicher Hilfsanzeigen wie folgt erstellen:

1. Wählen Sie den Befehl **Index** aus dem Menü **Hilfe** aus, und plazieren Sie den Cursor an dem Eintrag `DO...LOOP`. Rufen Sie anschließend die On-Line-Hilfe für diese Anweisung durch Drücken der Taste F1 auf.
2. Wählen Sie den Befehl **Drucken...** aus dem Menü **Datei** durch Betätigen der Tastenkombination ALT+D und der Taste D aus.

   Wenn das Hilfsfenster aktiv ist, kann anhand dieser Option entweder ein bestimmter ausgewählter Text oder die gesamte aktuelle Hilfsanzeige ausgedruckt werden (siehe Abbildung 7.4).

Abbildung 7.4 Das Dialogfeld Drucken (Hilfe)



3. Drucken Sie diese Hilfsanzeige durch Betätigen der EINGABETASTE aus, falls Ihnen ein Drucker zur Verfügung steht, oder löschen Sie das Dialogfeld über die ESC-TASTE. (Wenn Sie versuchen auszudrucken , ohne über einen Drucker zu verfügen, erscheint nach einer gewissen Zeit die Fehlermeldung `Laufzeitfehler am Gerät`.)
4. Wählen Sie die gewünschten Hyperverbindungen, und drucken Sie diese aus.
5. Schließen Sie das Hilfsfenster durch Betätigen der ESC-TASTE.

Informieren Sie sich über die Programmiermethoden anhand der ausgedruckten Hilfstextausgabe (oder der On-Line-Hilfe). Mit Hilfe der nächsten Arbeitsschritte werden Sie eine bedingungslose Schleife einfügen.

15. Code-Eingabe in QKARTEN

1. Bringen Sie den Cursor in die Zeile unterhalb dieses Kommentarblocks (ungefähr Zeile 175):

```
' Eine bedingungslose DO...LOOP-Anweisung beläßt die
' Maske fortwährend auf dem Bildschirm. Der DO-Teil der
' Anweisung steht in der nächsten Code-Zeile, während der
' LOOP-Teil unmittelbar vor der Anweisung END steht.
' Diese Schleife umfaßt die zentrale Logik, die dem
' Benutzer erlaubt, mit QKARTEN zu arbeiten.
```

2. Erstellen Sie eine Leerzeile über die EINGABETASTE.
3. Geben Sie das Wort `do` ein, und drücken Sie die EINGABETASTE.
4. Bringen Sie den Cursor in die Zeile unterhalb dieses Kommentars (ungefähr Zeile 220):

```
' Das ist das untere Ende der bedingungslosen DO-
' Schleife.
```

5. Erstellen Sie eine Leerzeile über die EINGABETASTE, und geben Sie das Wort `loop` ein.
6. Wählen Sie den Befehl **Start** aus dem Menü **Ausführen** aus.

   Sie werden die bedingungslose **DO**-Schleife unterbrechen müssen.
7. Drücken Sie STRG+UNTBR, um die Ausführung der bedingungslosen **DO**-Schleife auszusetzen.

   Während die Ausführung unterbrochen ist, können die Variablen mit Hilfe von Befehlen wie **Aktuellen Wert anzeigen** untersucht werden. Anschließend kann die Programmausführung durch Auswählen des Befehls **Weiter** aus dem Menü **Ausführen** (mit Hilfe der Tastenkombination ALT+A und der Taste W) fortgesetzt werden.
8. Speichern Sie das Programm unter dem Namen *kapitel7.bas*.

Weitere Informationen

Nähere Informationen zur On-Line-Hilfe sind folgenden Kapiteln zu entnehmen:

Kapitel	*Thema*
Abschnitt 10.8, "Verwendung der On-Line-Hilfe", und Kapitel 21, "Das Menü Hilfe"	Informationen zu den Befehlen des Hilfsmenüs und vollständige Beschreibung sämtlicher Eigenschaften der On-Line-Hilfe.

8 Verwendung des Beispielcodes aus der On-Line-Hilfe

In den vorhergehenden Kapiteln haben Sie zahlreiche Merkmale der On-Line-Hilfe von Microsoft QuickBASIC kennengelernt. Sowohl der Hilfebildschirm als auch die jeder Anweisung und Funktion entsprechenden Anzeige **Details** weisen jeweils eine Hyperverbindung zu einem oder mehreren Codebeispielen auf. In diesem Kapitel werden Sie Beispielcode aus der On-Line-Hilfe zur Verwendung in QKARTEN kopieren und bei folgenden Vorgängen einsetzen:

- Kopieren von Beispielcode aus der On-Line-Hilfe und Einfügen in das Arbeitsfenster.
- Bearbeiten und Anpassen der aus der On-Line-Hilfe kopierten Textpassagen in einem bestimmten Programm.

Dieses Kapitel erfordert ungefähr anderthalb Arbeitsstunden.

Kopieren von Beispielcode aus der On-Line-Hilfe

Beim Anfordern der On-Line-Hilfe für Schlüsselwörter beschreibt der Hilfebildschirm die Schlüsselwortaktion und -syntax. Wenn Ihnen die Anweisung bereits bekannt ist, genügt wahrscheinlich die im Syntaxblock vorhandene Information zum Schreiben des Codes. Wenn Ihnen dagegen die Anweisung unbekannt ist, verdeutlichen die Anzeigen **Details** und **Beispiel** die Anwendung der Syntax. Bei den nächsten Codeeingaben werden Sie in der On-Line-Hilfe ein Beispiel finden, das kopiert und in QKARTEN benutzt werden kann. Der Beispielcode aus der On-Line-Hilfe muß nicht immer für QKARTEN vollkommen geeignet sein. Ein Teil des kopierten Codes widerspricht dem Programmcode. Machen Sie sich jedoch keine Sorgen – diese Widersprüche werden im Laufe der Programmentwicklung ausgeglichen.

16. Code-Eingabe in QKARTEN

1. Positionieren Sie den Cursor in der nach folgendem Kommentarblock stehenden Zeile (ungefähr Zeile 188):

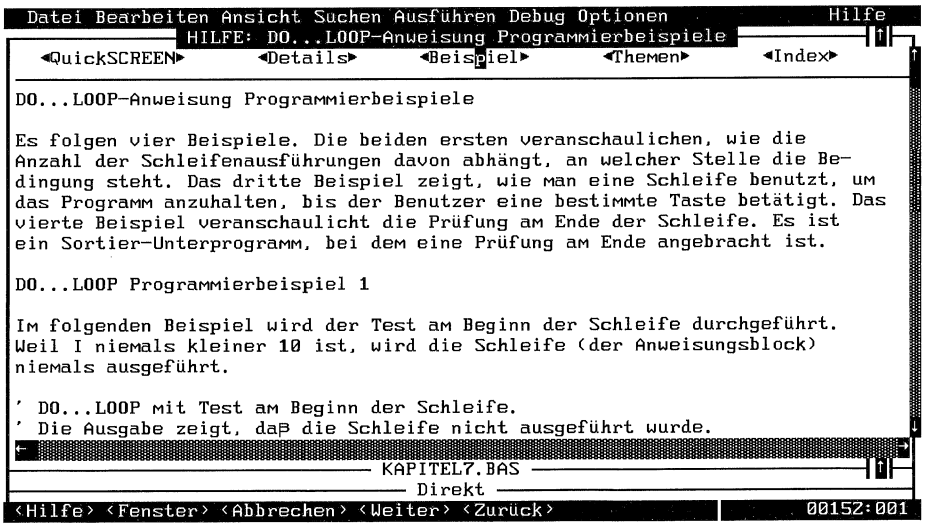
```
' Erfasse eine Tastenbetätigung des Benutzers mit Hilfe
' einer bedingten DO...LOOP-Anweisung und weise diese
' innerhalb der Schleife anhand der INKEY$-Funktion einer
' Zeichenkettenvariablen zu. Der WHILE-Teil der LOOP-
' Zeile testet die Zeichenkettenvariable fortlaufend.
' INKEY$ gibt solange eine durch "" dargestellte
' Nullzeichenkette (Länge Null) zurück, bis eine Taste
' betätigt wird. In diesem Fall gibt INKEY$ eine
' Zeichenkette zurück, deren Länge größer als Null ist,
' und die Schleife wird beendet.
```

2. Erstellen Sie eine Leerzeile mit der EINGABETASTE, und geben Sie do ein.
3. Fordern Sie die Hilfe für das Schlüsselwort **DO** durch Betätigen der Taste F1 an.

   Befindet sich der Cursor während der Betätigung von F1 auf einer Leerstelle, zeigt QuickBASIC eine kontextabhängige Hilfe für das Symbol auf der linken Seite des Cursors an. Es ist zu beachten, daß der Cursor im Arbeitsfenster bleibt.
4. Bringen Sie den Cursor durch zweifaches Betätigen der Taste F6 in das Hilfsfenster.
5. Wählen Sie die Hyperverbindung **Beispiel** durch Drücken der Tasten B und F1 aus.

   Im Hilfsfenster ist jetzt die Beispielsanzeige zu sehen (siehe Abbildung 8.1):

Abbildung 8.1 Beispielanzeige für die DO...LOOP-Anweisung



Der erste Absatz der Anzeige stellt die unten aufgeführten Beispiele vor. Die Beschreibung des dritten Beispiels entspricht genau der Aufgabe, die zu diesem Zeitpunkt in QKARTEN codiert werden soll. Kopieren Sie den Beispielcode durch folgende Arbeitsschritte in den Zwischenspeicher:

17. Code-Eingabe in QKARTEN

1. Gehen Sie zum 3. Beispiel, das wie folgt aussieht:

```
' DO...LOOP mit Test am Ende der Schleife.
DO
      Auswahl$ = INKEY$
LOOP WHILE Auswahl$ = ""
```

2. Markieren Sie die vier oben aufgeführten Zeilen.
3. Kopieren Sie mit dem Befehl **Kopieren** aus dem Menü **Bearbeiten** die Zeilen in den Zwischenspeicher (ALT+B und K).
4. Schließen Sie das Hilfsfenster mit ESC.
5. Markieren Sie das vorher im Arbeitsfenster eingegebene Wort `do` durch Betätigen der Tastenkombination UMSCHALT+POS1.
6. Wählen Sie den Befehl **Einfügen** aus dem Menü **Bearbeiten** durch Betätigen der Tastenkombination ALT+B und der Taste E.

   Der im Zwischenspeicher vorhandene Text ersetzt den markierten Text.
7. Betätigen Sie die Tastenkombination ALT+A und S (der Befehl **Start** aus dem Menü **Ausführen**), um nach möglichen Fehlern zu suchen, und kehren Sie anschließend zu QuickBASIC zurück, indem Sie die Tastenkombination STRG+UNTBR drücken.

Sie können mit Hilfe der On-Line-Hilfe nach Widersprüchen zwischen dem kopierten Code und QKARTEN suchen:

1. Positionieren Sie den Cursor auf `Auswahl$`, und drücken Sie F1.
2. Die On-Line-Hilfe zeigt, daß die Variable mit dem Namen `Auswahl$` jetzt zweimal in QKARTEN auftritt, einmal im Modul-Ebenen-Code und einmal in der Prozedur `ZeichkeBearb`. Diese Vorkommen widersprechen sich jedoch nicht, da zu einer Prozedur lokale Variablen, auf Modul-Ebene nicht bekannt sind und umgekehrt.

Einrückung eines Codeblocks

Die zuletzt eingegebene Schleife ist innerhalb der vorher (in Kapitel 7, "Programmieren mit der On-Line-Hilfe", Abschnitt "15. Code-Eingabe in QKARTEN") eingegebenen bedingungslosen Schleife "verschachtelt". Das bedeutet, daß diese Schleife bei jeder Ausführung der äußeren Schleife einmal ausgeführt wird. Verschachtelter Code wird üblicherweise um einen Tabulatorstopp nach rechts vom umfassenden Code eingerückt, um die Verschachtelungs-Ebenen des Programms übersichtlich darzustellen. Mehrere Textzeilen lassen sich blockweise wie folgt einrücken:

1. Wählen Sie die drei zuletzt eingegebenen Codezeilen aus:

```
DO
     Auswahl$ = INKEY$
LOOP WHILE Auswahl$ = ""
```

2. Schieben Sie den gesamten Block durch Betätigen der TAB-TASTE um einen Tabulatorstopp nach rechts.
3. Drücken Sie eine RICHTUNGSTASTE, um den Block wieder normal darzustellen.

Der restliche in QKARTEN einzugebende Code ist ebenfalls innerhalb der bedingungslosen **DO**-Schleife verschachtelt, so daß jeder Block gleichermaßen eingerückt werden sollte.

Kopieren umfangreicher Code-Blöcke aus der On-Line-Hilfe

Der im vorhergehenden Abschnitt kopierte Code war kurz und einfach. Da er nur einen einzigen Variablennamen enthielt, war die Wahrscheinlichkeit eines Widerspruches mit einem in QKARTEN bereits vorhandenen Symbol sehr gering.

Es können aber auch umfangreichere Beispielcodeblöcke aus der On-Line-Hilfe kopiert und im Programm eingefügt werden. In diesem Fall kann jedoch der kopierte Code mit bereits im Programm geschriebenem Code in Widerspruch stehen. In diesem Abschnitt werden Sie einen größeren Beispielcode in QKARTEN kopieren und diesen dem Programm erheblich anpassen. Die teilweise auftretenden Widersprüche werden zu einem späteren Zeitpunkt ausgeglichen.

18. Code-Eingabe in QKARTEN

1. Bringen Sie den Cursor in die Zeile unterhalb des nachstehenden Kommentarblocks (ungefähr Zeile 207) zurück:

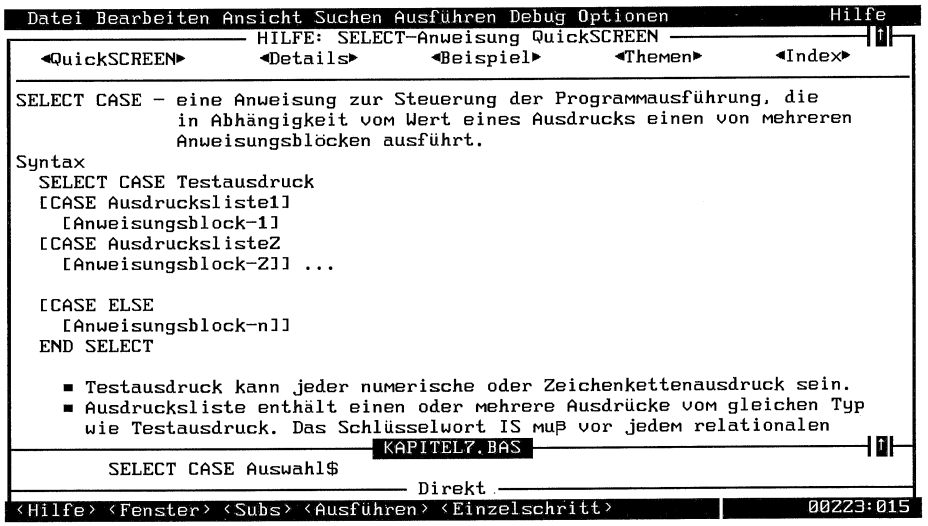
```
' Die ELSE-Klausel wird nur dann ausgeführt, wenn
' Auswahl$ länger als ein einziges Zeichen ist (und
' demnach keine Befehlszeilentaste darstellt). Wenn
' Auswahl$ keine ASCII-Taste ist, stellt sie eine
' "erweiterte" Taste dar. (QKARTEN sucht nach den
' erweiterten Tasten, da diese die RICHTUNGSTASTEN des
' numerischen Tastenblocks umfassen.) In diesem Fall wird
' die Funktion RIGHT$ zum Abschneiden des zusätzlichen
' Bytes eingesetzt, so daß der resultierende Wert
' möglicherweise dem Wert einer RICHTUNGSTASTE
' entspricht. Verwenden Sie eine SELECT CASE-
' Konstruktion, um entsprechend der betätigten
' RICHTUNGSTASTE des numerischen Tastenblocks im Programm
' fortzufahren.
```

2. Erstellen Sie eine Leerzeile mit der EINGABETASTE, und geben Sie folgendes ein:

```
select case Auswahl$
```

3. Drücken Sie die POS1-TASTE, und fordern Sie Hilfe für die `SELECT CASE`-Anweisung durch Betätigen der Taste F1 an.

Abbildung 8.2 Hilfebildschirm für die Anweisung SELECT CASE



In Situationen mit zahlreichen Eingabemöglichkeiten, die größtenteils eindeutig behandelt werden müssen, ist die **SELECT CASE**-Anweisung einer Schalttafel ähnlich. Es ist zwar nicht erstrebenswert, eine eindeutige Aktion für jede unterschiedliche Tastenbetätigung vorzusehen, jedoch erlaubt dieser Strukturtyp das Hinzufügen von **CASE**-Klauseln für jede Taste, die einzeln behandelt werden soll. Die restlichen Tasten können dann mit Hilfe der **CASE ELSE**-Anweisung als "nicht zu den oben erwähnten Kategorien gehörend" behandelt werden.

Mit den nächsten Arbeitsschritten kopieren Sie einen großen Codeteil aus der On-Line-Hilfe:

19. Code-Eingabe in QKARTEN

1. Bringen Sie den Cursor durch zweifaches Betätigen der Tastenkombination UMSCHALT+F6 in das Hilfsfenster.
2. Wählen Sie die Hyperverbindung **Beispiel** am oberen Ende der Anzeige durch Drücken der Tasten B und F1 aus.
3. Wählen Sie den Befehl **Suchen...** aus dem Menü **Suchen** (ALT+S und S) aus, geben Sie `2. Beispiel` im Textfeld ein, und drücken Sie die EINGABETASTE.
4. Gehen Sie 15 Zeilen nach unten in die Zeile: `IF LEN(Auswahl$) = 1 THEN`
5. Markieren Sie diese und die nächsten 43 Zeilen bis `END IF`. (Stellen Sie sicher, daß die abschließende `LOOP`-Anweisung *nicht* zu dem markierten Text gehört.)
6. Wählen Sie den Befehl **Kopieren** aus dem Menü **Bearbeiten** aus (ALT+B und K).
7. Schließen Sie das Hilfsfenster mit der ESC-TASTE, um den Cursor in die `SELECT CASE`-Anweisung im Arbeitsfenster zurückzubringen.
8. Markieren Sie die Zeile `SELECT CASE Auswahl$` durch Betätigen der Tastenkombination UMSCHALT+ENDE.
9. Führen Sie den Befehl **Einfügen** aus dem Menü **Bearbeiten** durch Betätigen der Tastenkombination ALT+B und der Taste E aus.

   Der nach dem Kommentarblock eingefügte Code müßte wie folgt aussehen:

```
IF LEN(Auswahl$) = 1 THEN
    ' Verarbeite ASCII-Tasten
       SELECT CASE ASC(Auswahl$)
           CASE ESC
              PRINT "Esc-Taste"
              END
           CASE IS < 32, 127
               PRINT "Steuercode"
           CASE 48 TO 57
```

```
                PRINT "Ziffer: "; Auswahl$
            CASE 65 TO 90
                PRINT "Großbuchstabe: "; Auswahl$
            CASE 97 TO 122
                PRINT "Kleinbuchstabe: "; Auswahl$
            CASE ELSE
                PRINT "Zeichensetzung: "; Auswahl$
            END SELECT
ELSE
    ' Wandle erweiterten 2-Byte Code in 1-Byte
    ' ASCII-Code um und verarbeite ihn
        Auswahl$ = RIGHT$(Auswahl$, 1)
        SELECT CASE Auswahl$
            CASE CHR$(UNTEN)
                PRINT "NACH UNTEN Pfeiltaste"
            CASE CHR$(AUF)
                PRINT "NACH OBEN Pfeiltaste"
            CASE CHR$(BILDAB)
                PRINT "BILD NACH UNTEN Taste"
            CASE CHR$(BILDOBEN)
                PRINT "BILD NACH OBEN Taste"
            CASE CHR$(POS1)
                PRINT "POS1-Taste"
            CASE CHR$(ENDE)
                PRINT "ENDE-Taste"
            CASE CHR$(RECHTS)
                PRINT "NACH RECHTS Pfeiltaste"
            CASE CHR$(LINKS)
                PRINT "NACH LINKS Pfeiltaste"
            CASE ELSE
                BEEP
        END SELECT
END IF
```

Testen Sie Ihre Arbeit, und speichern Sie diese anschließend:

20. Code-Eingabe in QKARTEN

1. Führen Sie den Befehl **Start** des Menüs **Ausführen** durch Betätigen der Tastenkombination ALT+A und der Taste S aus.

   Beim Betätigen der Tasten werden deren Bezeichnungen auf dem QKARTEN-Ausgabebildschirm ausgegeben.

2. Kehren Sie mit der ESC-TASTE in die QuickBASIC-Umgebung zurück.

   Das Programm wird durch den Fall `"ESC"` im kopierten Code beendet.

3. Speichern Sie das Programm unter dem Namen *kapitel8.bas*.

Dieser Beispielcode ist leicht an QKARTEN anzupassen, da die Behandlung der Tastatureingabe zu üblichen Aufgaben gehört. Dieser Code kann jederzeit kopiert und dann dem jeweiligen Programm angepaßt werden.

Bearbeiten des aus der On-Line-Hilfe kopierten Codeblocks

Die Analyse des aus anderen Programmen stammenden Codes bildet die beste Möglichkeit zur Verbesserung der eigenen Programmiertechnik. Das Verändern von Code aus der On-Line-Hilfe und dessen Einsatz in den eigenen Programmen kann sich auch als zeitsparend erweisen. Jeder Fall aus dem Block `SELECT CASE...END SELECT` umfaßt einen Tasten-Code, der einen vom Benutzer in QKARTEN auszuwählenden Befehl darstellt. Dieser Block läßt sich anhand nachstehender Arbeitsschritte ändern:

21. Code-Eingabe in QKARTEN

1. Markieren Sie den gesamten ersten Block, der sich unterhalb des Kommentars `' Verarbeite ASCII-Tasten` (ungefähr Zeile 221) befindet:

```
SELECT CASE ASC(Auswahl$)
    CASE ESC
        PRINT "Esc-Taste"
        END
    CASE IS < 32, 127
        PRINT "Steuercode"
    CASE 48 TO 57
        PRINT "Ziffer: "; Auswahl$
    CASE 65 TO 90
        PRINT "Großbuchstabe: "; Auswahl$
    CASE 97 TO 122
        PRINT "Kleinbuchstabe: "; Auswahl$
    CASE ELSE
        PRINT "Zeichensetzung "; Auswahl$
END SELECT
```

   Dieser Block zeigt den Typ der vom Benutzer betätigten ASCII-Tasten an. QKARTEN verfügt bereits über eine Prozedur mit dem Namen `AsciiTast`, die den Code der betätigten Taste untersucht und andere Prozeduren zur Durchführung der Befehle aufruft.

Der im nächsten Arbeitsschritt eingegebene Aufruf der Prozedur `AsciiTast` ersetzt den ausgewählten Block.

2. Geben Sie folgende Zeile an der Cursorstelle ein:

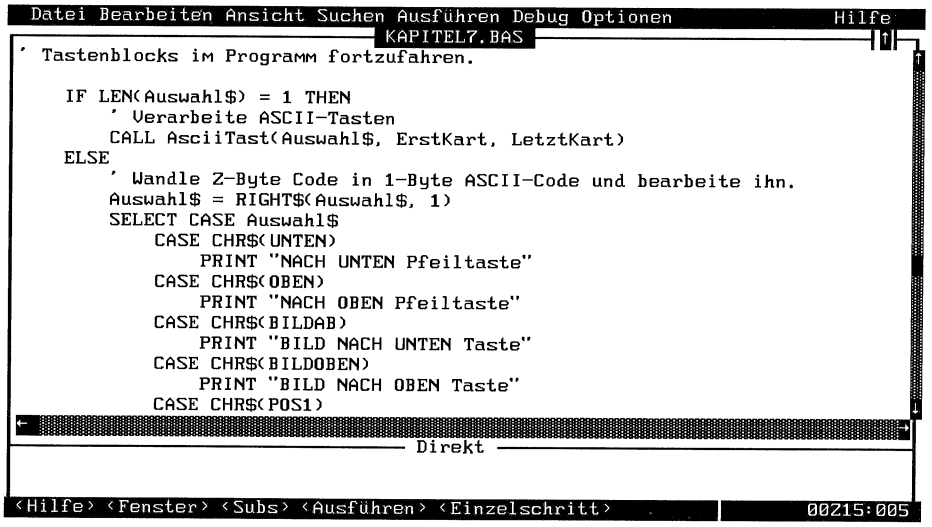
   ```
   call AsciiTast(Auswahl$, ErstKart, LetztKart)
   ```

   Die Prozedur `AsciiTast` wird mit drei Argumenten aufgerufen, die die zur Durchführung ihrer Aufgaben erforderlichen Informationen erteilen. `Auswahl$` enthält den Code der betätigten Taste, `ErstKart` die Nummer der im Vordergrund der Anzeige vorhandenen Karte und `LetztKart` die Nummer des letzten Datensatzes im Datenfeld `Index()`.

3. Rücken Sie die eingegebene Zeile zwei Tabulatorstopps ein.

   Der bearbeitete Code müßte wie folgt aussehen (siehe Abbildung 8.3):

Abbildung 8.3 Aufruf der Prozedur AsciiTast



4. Führen Sie den Befehl **Start** aus dem Menü **Ausführen** aus (ALT+A und S).

Versuchen Sie, einige QKARTEN-Befehle auszuführen. Sie sind jetzt in der Lage, die Karte im Vordergrund der Anzeige zu bearbeiten. Wenn Sie jedoch eine der RICHTUNGSTASTEN betätigen, erscheint immer noch nur eine Beschreibung der Taste, wobei die QKARTEN-Anzeige überschrieben wird.

Damit die RICHTUNGSTASTEN in QKARTEN funktionieren, müssen weitere aus der On-Line-Hilfe kopierten Textpassagen bearbeitet werden:

22. Code-Eingabe in QKARTEN

1. Drücken Sie die Taste B, um den QKARTEN-Befehl **Beenden** auszuwählen und zu QuickBASIC zurückzukehren.
2. Positionieren Sie den Cursor in der ersten Spalte der Zeile `PRINT "NACH UNTEN Pfeiltaste"`.
3. Markieren Sie die gesamte Zeile durch Betätigen der Tastenkombination UMSCHALT+ENDE, und ersetzen Sie sie durch folgende Zeile:

   ```
   ErstKart = ErstKart - 1
   ```

   Diese Zeile wird ausgeführt, wenn der Benutzer die NACH UNTEN TASTE betätigt, und vermindert den Wert der Karte, die die Daten im Vordergrund der Anzeige darstellt, um eins. Auf diese Weise weist der Benutzer das Programm QKARTEN an, den Datensatz zu zeigen, dessen Nummer um eins kleiner ist als die der aktuellen ersten Karte.
4. Rücken Sie die Zeile ein, und bearbeiten Sie die restlichen Fälle, bis der endgültige Block genau wie unten angegeben aussieht:

   ```
   SELECT CASE Auswahl$
              CASE CHR$(NACH UNTEN)
                   ErstKart = ErstKart - 1
              CASE CHR$(NACH OBEN)
                   ErstKart = ErstKart + 1
              CASE CHR$(BILD NACH UNTEN)
                   ErstKart = ErstKart - KPROANZEIGE
                CASE CHR$(BILD NACH OBEN)
                   ErstKart = ErstKart + KPROANZEIGE
              CASE CHR$(POS1)
                   ErstKart = LetztKart
              CASE CHR$(ENDE)
                   ErstKart = 1
              CASE ELSE
                   BEEP
          END SELECT
   ```

   Die letzten zwei aus der On-Line-Hilfe kopierten Fälle für die RICHTUNGSTASTEN (`NACH RECHTS` und `NACH LINKS`) des Blocks werden in QKARTEN nicht benötigt, und deshalb gelöscht.

   Der Anweisungsteil `CASE ELSE` wird ausgeführt, wenn der Benutzer von QKARTEN eine Taste mit erweitertem Zeichencode betätigt, für die kein Fall vorgesehen ist.
5. Speichern Sie das Programm unter dem Namen *kapitel8.bas*.

Die im Fall **CASE...ELSE** aus dem Hilfsbeispiel verwendete `BEEP`-Anweisung kann durch die in Kapitel 5, "Das Programm QKARTEN", erstellte Prozedur `Alarm` ersetzt werden. In den nächsten Arbeitsschritten wird der Befehl **Suchen** zum Kopieren eines früheren Aufrufes der Prozedur `Alarm` in den Zwischenspeicher verwendet. Markieren Sie anschließend die `BEEP`-Anweisung und ersetzen sie durch `CALL Alarm`.

4. optionale Code-Eingabe in QKARTEN

1. Wählen Sie den Befehl **Suchen...** aus dem Menü **Suchen** durch Betätigen der Tastenkombination ALT+S und der Taste S aus, geben Sie `CALL Alarm` im Textfeld ein. Drücken Sie die EINGABETASTE.
2. QuickBASIC sucht und markiert die in Kapitel 5, "Das Programm QKARTEN", eingegebene Prozedur `CALL Alarm`.
3. Führen Sie den Befehl **Ausschneiden** aus dem Menü **Bearbeiten** durch Betätigen der Tastenkombination ALT+B und der Taste A aus. (Die Worte `CALL Alarm` brauchen nicht mehr hervorgehoben zu werden, da sie bereits markiert sind.)
4. Wählen Sie den Befehl **Suchen...** erneut aus, geben Sie `BEEP` im Textfeld ein und drücken Sie die EINGABETASTE.

   Der Suchbegriff wird in der Anweisung `CASE ELSE` gefunden und markiert.
5. Führen Sie den Befehl **Einfügen** aus dem Menü **Bearbeiten** aus, um `BEEP` mit `CALL Alarm` zu ersetzen.

Abschluß des QKARTEN-Codes

Wenn der Benutzer die RICHTUNGSTASTEN betätigt, ändert der **SELECT CASE**-Block den Wert von `ErstKart`, ohne jedoch die in den Karten angezeigte Information zu ändern.

Der folgende Code verarbeitet drei Möglichkeiten, die sich aus bestimmten Kombinationen der RICHTUNGSTASTEN ergeben können, und zeigt die korrekten Daten in den Karten an:

23. Code-Eingabe in QKARTEN

1. Positionieren Sie den Cursor in der Zeile unterhalb des folgenden Kommentars (ungefähr Zeile 250):

```
' Ordne die Karten gemäß der vom Benutzer betätigten
' Taste an, und rufe dann die Prozedur ZeigKart auf, um
' den Stapel in der neuen Anordnung anzuzeigen.
```

2. Erstellen Sie eine Leerzeile mit der EINGABETASTE, und geben Sie den folgenden Code genau wie angegeben ein:

```
if ErstKart < 1 then ErstKart = LetztKart + ErstKart
if ErstKart > LetztKart then ErstKart = ErstKart - _
LetztKart
if ErstKart <= 0 then ErstKart = 1
```

   Diese drei Zeilen "rotieren" den Kartenstapel mit Hilfe der einzeiligen **IF...THEN**-Anweisung. Diese Zeilen verhindern die Übergabe von `ErstKart` an `ZeigKart`, wenn der Wert der Karte außerhalb des Datenfeldbereichs von `Index()` liegt.

3. Geben Sie jetzt die abschließende Codezeile ein:

```
CALL ZeigKart(ErstKart, LetztKart)
```

   Die Prozedur `ZeigKart` setzt die richtigen Daten in die Vordergrundkarte und die richtige Bezeichnung in sämtliche darauffolgende Karten ein.

4. Speichern Sie das Programm unter dem Namen *kapitel8.bas*.

5. Zur Ausführung des Programms QKARTEN wählen Sie den Befehl **Start** aus dem Menü **Ausführen** durch Betätigen der Tastenkombination ALT+A und der Taste S aus.

Verwendung des Programms QKARTEN

Die auf der Befehlszeile am unteren Ende des QKARTEN-Ausgabebildschirms eingeblendeten Befehle sind jetzt voll einsatzfähig und können innerhalb weniger Minuten getestet werden. Die Befehle sind in nachstehender Liste zusammengefaßt:

QKARTEN-Befehl	*Ergebnis*
Bearbeiten	Erlaubt die Änderung jedes zu einer Karte im Vordergrund der Anzeige gehörenden Feldes.
Hinzufügen	Setzt eine "leere Karte" zum Ausfüllen in den Vordergrund der Anzeige.
Kopieren	Dupliziert sämtliche Felder der aktuellen oberen Karte in eine neue Karte.
Löschen	Die aktuelle obere Karte wird markiert und beim Auswählen des Befehls **Beenden** gelöscht. Wenn Sie die Karte nicht mehr löschen wollen, können Sie das Programm über die Tastenkombination STRG+UNTBR unterbrechen, um die Karte beizubehalten.

QKARTEN-Befehl	*Ergebnis*
Suchen	Erlaubt es Ihnen, zu einem der Felder zu gehen und einen Suchbegriff einzugeben. Wird dieser anschließend gefunden, wird die entsprechende Karte an erster Stelle plaziert. Wenn kein genau übereinstimmender Suchbegriff gefunden wird, gibt QKARTEN einen Piepton aus.
Sortieren	Ordnet die Karten in der alphabetischen (oder numerischen) Reihenfolge des Feldes, in dem sich der Cursor befindet. Wenn sich der Cursor beispielsweise im Postleitzahlfeld **PLZ** befindet, ordnet **Sortieren** die Karten in steigender Reihenfolge der Postleitzahlen an. Wenn der Cursor im Feld **Land** steht, werden die Karten in alphabetischer Reihenfolge der Länder angeordnet.
Drucken	Druckt nur die Feldnamen- und adressen in der Form eines Postetiketts aus.
Ende	Schreibt das laufende Index-Datenfeld in die Direktzugriffsdatei. Wenn Karten zum Löschen markiert sind, werden diese aus der Datei entfernt. Beim Verlassen von QKARTEN werden auch die angegebenen Änderungen und Ergänzungen gespeichert. Anschließend beendet der Befehl **Beenden** das Programm QKARTEN.

Testen Sie auch die RICHTUNGSTASTEN. Die meisten funktionieren zwar, Sie haben jedoch beim Kopieren des Codes aus der On-Line-Hilfe versehentlich eine Unstimmigkeit in den Bezeichnungen miteingeschlossen, die einen Fehler im Programm verursacht. Im nächsten Kapitel werden Sie den Fehler mit Hilfe der QuickBASIC-Fehlerbeseitigung bereinigen.

Weitere Informationen

Nähere Informationen über die On-Line-Hilfe sind folgenden Abschnitten zu entnehmen:

Kapitel	*Thema*
Abschnitt 10.8, "Verwendung der On-Line-Hilfe", und Kapitel 21, "Das Menü Hilfe"	Informationen über die Befehle des QuickBASIC-Hilfsmenüs und vollständige Beschreibung aller Eigenschaften der On-Line-Hilfe.

9 Fehlerbeseitigung während der Programmierung

Der intelligente Editor von Microsoft QuickBASIC stellt sicher, daß keine Syntaxfehler bei der Eingabe von Anweisungen vorkommen. Andere in QuickBASIC verfügbare Einrichtungen sind bei der Entdeckung von Fehlern in der allgemeinen Logik oder in bestimmten Programmteilen, die das Verhalten des Programms unvorhersehbar gestalten können, behilflich. Einige dieser Einrichtungen sind im Menü **Debug** vorhanden, andere befinden sich im Menü **Ausführen** oder werden vollständig über Funktionstasten gesteuert. Sie erlauben Ihnen, die Programmausführung jederzeit anzuhalten, die vom Programm auszuführenden Anweisungen zu steuern und die Variablenwerte zu überwachen.

In diesem Kapitel werden Sie einen Fehler finden und beseitigen. Die Übung umfaßt:

- die Verwendung der Fehlerbeseitigung (Debuggen) von QuickBASIC.
- die Definition und den Aufruf einer mit Parametern versehenen Prozedur.
- die Erstellung einer selbständigen ausführbaren Datei aus QuickBASIC.

Dieses Kapitel erfordert ungefähr anderthalb Arbeitsstunden.

Befehle zum Debuggen

Die meisten QuickBASIC-Befehle zum Debuggen befinden sich im Menü **Debug**. Die gebräuchlichsten Einrichtungen gehören jedoch zum Menü **Ausführen** oder sind über Funktionstasten zugänglich.

Die Befehle des Menüs Debug

Die Befehle des Menüs **Debug** ermöglichen die Steuerung des Programmablaufs und die Überwachung der Variablenwerte in einem laufenden Programm.

Befehl (und direkte Tastenkombination)	*Aktion*
Variable anzeigen	Setzt eine gegebene Variable im Debug-Fenster ein und aktualisiert deren Wert fortlaufend.
Aktuellen Wert anzeigen (UMSCHALT+F9)	Gibt den aktuellen Wert einer markierten Variablen an; setzt die Variable wahlweise im Debug-Fenster ein.
Anzeigevariable löschen	Löscht eine gegebene Variable aus dem Debug-Fenster.
Haltepunkt ein/aus (F9)	Setzt, falls erforderlich, einen "Haltepunkt" an der Cursorstelle. Der Haltepunkt bezeichnet eine bestimmte Stelle, an der die Programmausführung unterbrochen wird. Wenn ein Haltepunkt an der Cursorstelle festgelegt ist, kann er durch Auswählen des Befehls **Haltepunkt ein/aus** ausgeschaltet werden.
Alle Haltepunkte löschen	Schaltet sämtliche Haltepunkte im Programm aus.

Die Befehle zum Debuggen im Menü Ausführen

Während der Fehlerbeseitigung wird die Programmausführung durch Befehle aus dem Menü **Ausführen** gesteuert.

Befehl (und direkte Tastenkombination)	*Aktion*
Start (UMSCHALT+F5)	Führt das Programm ab der ersten Anweisung im Arbeitsfenster aus.
Neustart	Setzt alle Variablen auf Null (oder auf Zeichenketten der Länge Null) zurück, kompiliert Vereinbarungen und legt die erste ausführbare Anweisung als aktuelle Anweisung fest.
Weiter (F5)	Setzt die Ausführung eines unterbrochenen Programms ab der aktuellen Anweisung fort. In einem noch nicht ausgeführten Programm hat dieser Befehl dieselbe Wirkung wie **Start**.

Bei der Fehlerbeseitigung verwendete Funktionstasten

Nachstehende Tabelle listet die über Funktionstasten gesteuerten Befehle zum Debuggen auf:

Befehl	*Aktion*
F7	Führt das Programm bis zu der Anweisung an der Cursorstelle aus.
F8	Beim Betätigen der Taste wird jeweils eine Anweisung ausgeführt. Besteht die aktuelle Anweisung aus einem Prozeduraufruf, bringt F8 die Prozedurdefinition in das Arbeitsfenster. Wenn das Programm noch nicht ausgeführt wurde, hat das Betätigen der Taste F8 dieselbe Wirkung wie der Befehl **Neustart** aus dem Menü **Ausführen**, das bedeutet, daß das Programm erneut gestartet und die erste ausführbare Anweisung als aktuelle Anweisung festgelegt wird. In einem unterbrochenen Programm führt F8 die aktuelle Anweisung in der Ausführungsfolge aus.
F10	Führt das Programm im Einzelschrittmodus aus. F10 übergeht sämtliche Anweisungen einer Prozedur oder führt sie wie eine einzige Anweisung aus, ohne die Prozedur im Arbeitsfenster anzuzeigen. Im übrigen hat F10 dieselbe Wirkung wie F8.

Debuggen einer Prozedur

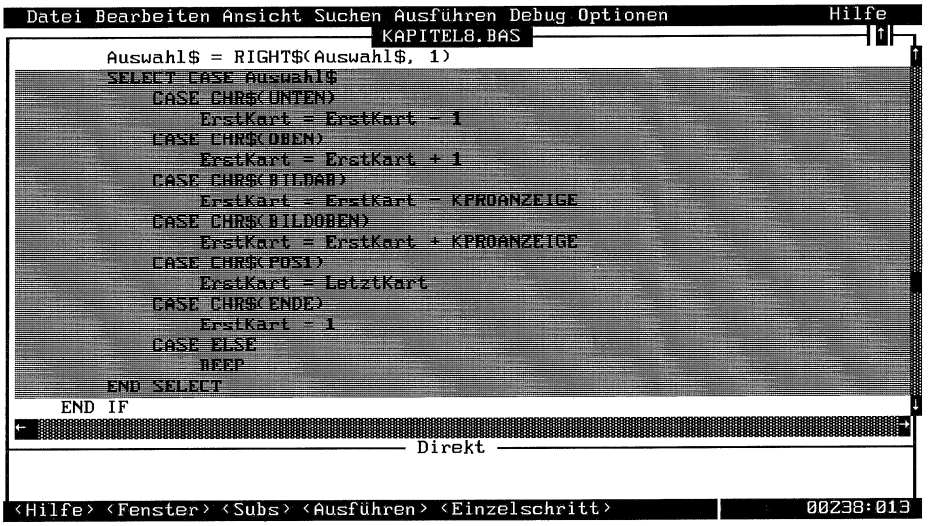
In diesem Abschnitt werden Sie alle bereits erworbenen Kenntnisse einsetzen, um den `SELECT CASE`-Block, der sich mit den RICHTUNGSTASTEN befaßt, in eine Prozedur umzuwandeln. Kapitel 5, "Das Programm QKARTEN", zeigt, wie einfach eine Prozedur in QuickBASIC zu definieren ist. Die in der nächsten Code-Zugangsfolge erstellte Prozedur wird durch Parameter definiert und mit Argumenten aufgerufen.

Die folgenden Arbeitsschritte erklären die Erstellung einer **SUB**-Prozedur mit dem Namen `RichtungsTaste`. Wie bereits in Kapitel 5 erwähnt, öffnet der intelligente Editor von QuickBASIC beim Verarbeiten des von einem Prozedurnamen gefolgten Wortes `sub` ein Fenster und beginnt die Prozedur mit den Anweisungen **SUB** und **END SUB**.

24. Code-Eingabe in QKARTEN

1. Starten Sie QuickBASIC, falls erforderlich, und laden Sie *kapitel8.bas*.
2. Positionieren Sie den Cursor in die äußerste linke Spalte der Zeile, die die Worte `SELECT CASE Auswahl$` enthält (ungefähr Zeile 224), und markieren Sie den gesamten 16-zeiligen Block einschließlich `END SELECT` (siehe Abbildung 9.1):

Abbildung 9.1 Auswahl des Blockes SELECT CASE Auswahl$



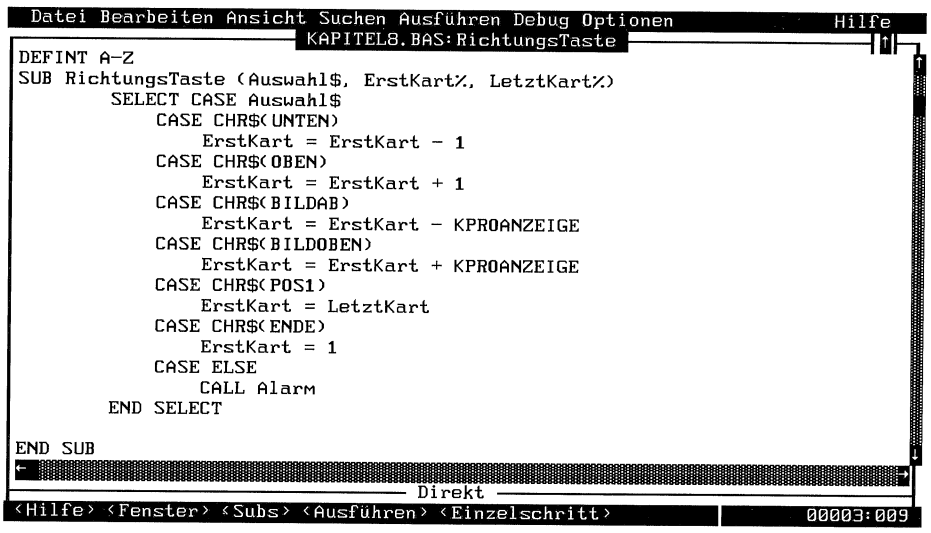
3. Wählen Sie den Befehl **Ausschneiden** aus dem Menü **Bearbeiten** aus, um den Text durch Betätigen der Tastenkombination ALT+B und A zu löschen und in den Zwischenspeicher zu kopieren.
4. Erstellen Sie eine Leerzeile mit der EINGABETASTE, und bringen Sie den Cursor in die erste Spalte der neuen Leerzeile (oberhalb der Anweisung `END IF`).
5. Geben Sie folgende Zeile ein

   ```
   sub RichtungsTaste (Auswahl$, ErstKart%, LetztKart%)
   ```

   und drücken Sie die EINGABETASTE.

   Wie Sie feststellen können, hat QuickBASIC eine **SUB...END SUB**-Prozedur mit dem angegebenen Namen begonnen. Sie befindet sich jetzt im Arbeitsfenster.
6. Bringen Sie den Cursor durch Betätigen der NACH UNTEN TASTE in die Leerzeile oberhalb der Anweisung `END SUB`.
7. Wählen Sie den Befehl **Einfügen** aus dem Menü **Bearbeiten** aus, um den Textblock aus dem Zwischenspeicher einzufügen. Die Prozedur sollte jetzt wie in Abbildung 9.2 aussehen.

Abbildung 9.2 Die Prozedur Richtungstaste im Arbeitsfenster



Nach Definition der Prozedur, ist eine **CALL**-Anweisung an der Stelle des gelöschten Texts wie folgt einzugeben, um die Anweisungen der **SUB**-Prozeduren in die Ausführungsfolge des Programms einzugliedern:

25. Code-Eingabe in QKARTEN

1. Wählen Sie den Befehl **SUBs** aus dem Menü **Ansicht** durch Betätigen der Taste F2 aus.

   Beachten Sie, daß `RichtungsTaste` den zehnten Eintrag nach *kapitel8.bas* darstellt.

2. Wählen Sie, falls erforderlich, *kapitel8.bas* aus. Bringen Sie den Cursor über die EINGABETASTE zum Modul-Ebenen-Code zurück.

3. Positionieren Sie den Cursor an das Ende der nachstehenden Zeile (ungefähr Zeile 222):

   ```
   Auswahl$ = RIGHT$ (Auswahl$, 1)
   ```

4. Erstellen Sie eine Leerzeile mit der EINGABETASTE und geben Sie folgende Zeile an der Cursorstelle ein, *ohne* die EINGABETASTE zu drücken:

   ```
   call RichtungsTaste (Auswahl$, ErstKart, LetztKart)
   ```

Die Prozedur `RichtungsTaste` ist komplizierter als `Alarm`, da sie anhand von Parametern (die in der vorher erstellten `SUB`-Anweisung in Klammern stehenden Worte) definiert und anhand von Argumenten (die eingeklammerten Worte in der `CALL`-Anweisung) aufgerufen wird. Die Bezeichnungen der Parameter müssen nicht, können jedoch mit denen der Argumente übereinstimmen.

Mit Hilfe der Prozedur `RichtungsTaste` ist der Modul-Ebenen-Code von QKARTEN leichter zu überschauen, da die Arithmetik der RICHTUNGSTASTEN durch eine einzige Anweisung, anstelle der 15 Anweisungszeilen, dargestellt wird. Die Anwendung von Prozeduren beeinträchtigt weder Ausführungsgeschwindigkeit noch Programmgröße, so daß die bessere Lesbarkeit des Programms den Programmieraufwand wert ist. Prozeduren eignen sich ganz besonders für Aufgaben, die sich im Programm häufig wiederholen.

Kennenlernen der Prozeduren

Sie können Prozeduren mit Hilfe der QuickBASIC-Fehlerbeseitigung kennenlernen. Zu diesem Zweck bewegen Sie den Cursor zum zuvor eingegebenen Aufruf von `RichtungsTaste` (ungefähr Zeile 226):

1. Führen Sie das Programm durch Drücken der Taste F7 bis zum Aufruf von `RichtungsTaste` aus.

   Während der Ausführung von QKARTEN werden Sie im folgenden Arbeitsschritt eine RICHTUNGSTASTE drücken, um die Ausführung beim Aufruf von `RichtungsTaste` zu unterbrechen.

2. Drücken Sie eine RICHTUNGSTASTE (vorzugsweise BILD NACH OBEN).

   Dadurch wird die Kontrolle an die QuickBASIC-Umgebung übergeben und die Anweisung `CALL Richtungstaste`, die noch nicht ausgeführt wurde und die aktuelle Anweisung darstellt, hervorgehoben.

3. Führen Sie die Prozedur `CALL RichtungsTaste` durch Betätigen der Taste F10 als eine einzige Anweisung aus. Anschließend wird die Anweisung `END IF` hervorgehoben, die jetzt die aktuelle Anweisung darstellt.

Fortsetzung eines unterbrochenen Programms

Ein Programm muß nach Durchführung von Änderungen nicht immer von Anfang an neu gestartet werden. Normalerweise genügt es, den Befehl **Weiter** aus dem Menü **Ausführen** auszuwählen, um die Ausführung von QKARTEN ab der Unterbrechung unverändert fortzusetzen. Beim Betätigen der RICHTUNGSTASTEN, müßten sich die auf den Karten vorhandenen Daten ändern. Zu diesem Zweck verfahren Sie wie folgt:

1. Setzen Sie die Programmausführung über die Taste F5 fort.
2. Drücken Sie die zum numerischen Tastenblock gehörende ENDE-TASTE, um die auf den Karten vorhandenen Daten in den letzten Eintrag einzusetzen. Sie werden feststellen, daß die ENDE-TASTE nicht funktioniert. Wenn Sie die in Kapitel 8 beschriebene optionale Code-Eingabe durchgeführt haben, gibt die Prozedur `Alarm` einen Ton aus, da die ENDE-TASTE nicht als zu dem `SELECT CASE`-Block gehörend erkannt wird. (Wurde der optionale Code nicht eingegeben, gibt QKARTEN einen Piepton aus, der auf einen Fehler hinweist).
3. Drücken Sie die Tasten NACH OBEN, NACH UNTEN, BILD NACH UNTEN und BILD NACH OBEN, die normalerweise alle funktionieren sollten. Das bedeutet, daß sich der Fehler im Code für die ENDE-TASTE befindet.
4. Kehren Sie zu QuickBASIC durch Betätigen der Tastenkombination STRG+UNTBR zurück.

   Nach der Rückkehr in die Umgebung, befindet sich der Cursor bei der Anweisung `Auswahl$ = INKEY$` oder an der Anweisung `LOOP WHILE Auswahl$ = ""`, je nach der während der Betätigung von STRG+UNTBR ausgeführten Anweisung. Sie können im Moment aus dieser `DO`-Schleife nicht herauskommen, und Sie können sie auch nicht mit Hilfe der Tasten F8 oder F10 verlassen. Sie müssen den Cursor auf die nach dieser Schleife stehende Anweisung positionieren und das Programm der Taste F7 bis zu diesem Punkt ausführen.
5. Bewegen Sie den Cursor ungefähr 27 Zeilen nach unten bis zu der folgenden Zeile (ungefähr Zeile 217), die nach dem großen Kommentarblock steht:

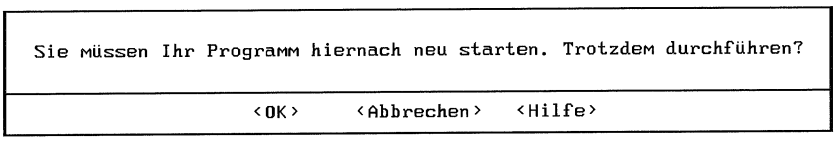
   ```
   IF LEN(Auswahl$) = 1 THEN
   ```
6. Führen Sie das Programm bis zu der Anweisung durch Drücken der Taste F7 aus. Anschließend erscheint die QKARTEN-Anzeige.
7. Drücken Sie erneut die ENDE-TASTE.

   QuickBASIC kehrt in die Umgebung zurück, wobei die Zeile hervorgehoben wird, in der sich der Cursor während der Betätigung der Taste F7 befand:

   ```
   IF LEN (Auswahl$) = 1 THEN
   ```

Eine unterbrochene Programmausführung kann normalerweise durch Betätigen der Taste F5 ab der Unterbrechung fortgesetzt werden. Es kann jedoch vorkommen, daß einige an dem unterbrochenen Programm vorgenommene Änderungen die einfache Fortsetzung verhindern. Bei den nächsten Eingaben in QKARTEN zeigt QuickBASIC eventuell ein Dialogfeld, mit dem Hinweis, daß die Änderungen einen Neustart des Programms erfordern (siehe Abbildung 9.3), an. Diese Meldung ist normal und Sie können die Bearbeitung nach Drücken der EINGABETASTE fortsetzen.

Abbildung 9.3 Die Fehlermeldung Programm-Neustart



Isolieren eines Fehlers

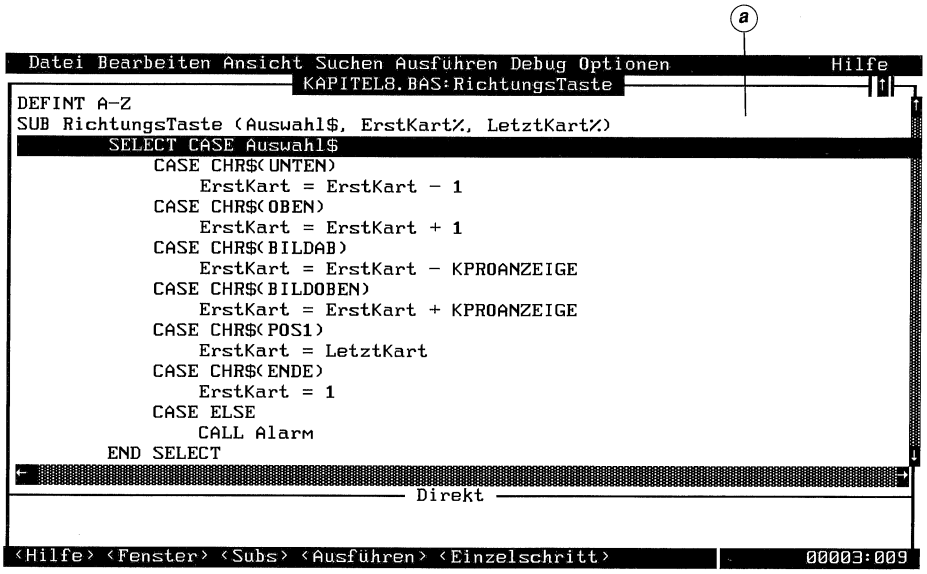
Mit der Fehlerbeseitigung von QuickBASIC sind Code- und Logikfehler leicht zu verfolgen und zu verbessern, da die Debug-Eigenschaften zum Verständnis der Funktionsweise jeder Code-Zeile beitragen. In der nächsten Eingabe in QKARTEN werden Sie den Fehler mit Hilfe des Einzelschrittmodus finden. Das Programm wurde durch eine RICHTUNGSTASTE unterbrochen, und es ist daher vorauszusehen, daß die Ausführung nach Verarbeitung der an der Cursorstelle stehenden Anweisung ab dem `ELSE`-Block fortgesetzt wird.

Sie können den Fehler wie folgt erkennen:

26. Code-Eingabe in QKARTEN

1. Führen Sie `IF` und die darauffolgende Anweisung aus, und bringen Sie den Cursor zu dem Aufruf von `RichtungsTaste` (betätigen Sie zweimal F8).
2. Bewegen Sie sich schrittweise durch Drücken der Taste F8 innerhalb der Prozedur `RichtungsTaste`, bis Sie sich bei der Anweisung `SELECT CASE` befinden.
3. Setzen Sie einen Haltepunkt durch Betätigen der Taste F9 in der aktuellen Zeile (siehe Abbildung 9.4).

Abbildung 9.4 Festlegung eines Haltepunktes



a) Haltepunkt.

4. Setzen Sie die Ausführung von QKARTEN durch zweimaliges Betätigen der Taste F5 fort. Nach Erscheinen des QKARTEN-Ausgabebildschirms, drücken Sie die ENDE-TASTE, um den vermutlichen Fehler auszulösen.

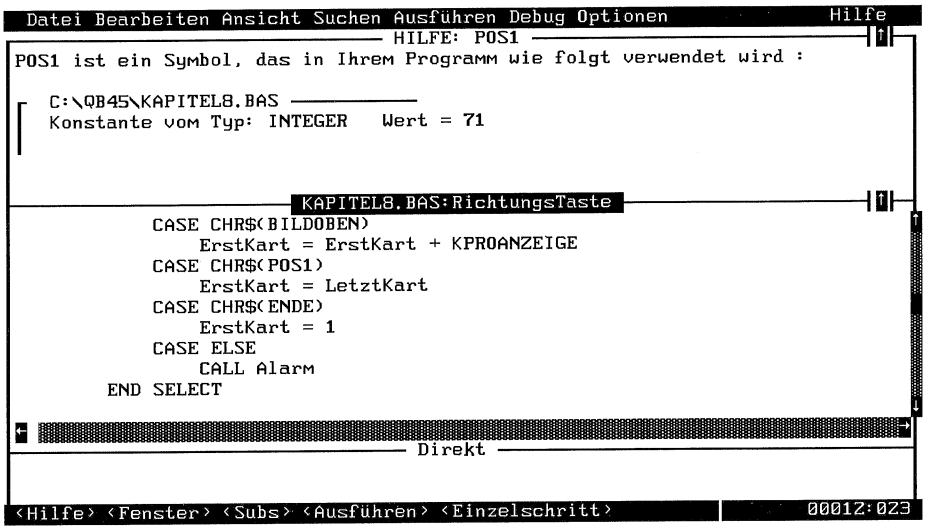
QuickBASIC bringt den Cursor in die mit dem Haltepunkt versehene Zeile zurück.

5. Bewegen Sie sich im Einzelschrittmodus mit Hilfe der Taste F8 bis zu der Anweisung `CASE ELSE`. Die Anweisung `CALL Alarm` wird invers dargestellt.

6. Positionieren Sie den Cursor innerhalb des Wortes `POS1` in der Anweisung `CASE CHR$ (POS1)`, und fordern Sie Hilfe durch Drücken der Taste F1 an.

Das Hilfsfenster wird geöffnet und beschreibt `POS1` als eine symbolische Konstante mit dem Wert `71` (siehe Abbildung 9.5).

Abbildung 9.5 Symbolhilfe für die symbolische Konstante POS1



7. Bewegen Sie den Cursor nach unten in die Anweisung `CASE CHR$ (ENDE-TASTE)`, positionieren Sie ihn auf das Wort `ENDE-TASTE` und fordern Sie Hilfe durch Drücken der Taste F1 an.

   Das Hilfsfenster wird geöffnet und beschreibt `ENDE-TASTE` als eine Variable des Typs `INTEGER`, obwohl sie eine symbolische Konstante wie `POS1` sein sollte.

8. Wählen Sie den Befehl **Aktuellen Wert anzeigen** aus dem Menü **Debug** aus.

   Der Wert von `ENDE-TASTE` wird mit Null angezeigt.

9. Setzen Sie `ENDE-TASTE` mit Hilfe der EINGABETASTE im Debug-Fenster ein.

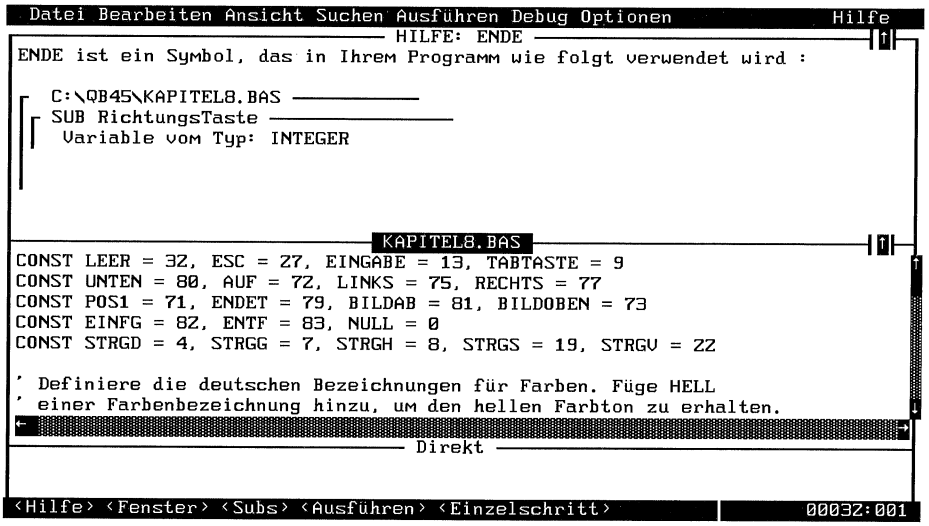
`ENDE-TASTE` sollte eine symbolische Konstante im Programm darstellen, die genau wie `POS1` im Vereinbarungs- und Definitionsabschnitt des Programms durch eine **CONST**-Anweisung definiert wird. Versuchen Sie folgendes:

27. Code-Eingabe in QKARTEN

1. Wählen Sie den Befehl **SUBs** aus dem Menü **Ansicht** (ALT+N und S) und dann den ersten Eintrag (*kapitel8.bas*) des Verzeichnisfeldes aus, um den Modul-Ebenen-Code des Programms im Arbeitsfenster anzuzeigen.
2. Positionieren Sie den Cursor über die Tastenkombination STRG+POS1 an den Anfang des Modul-Ebenen-Codes.
3. Wählen Sie den Befehl **Suchen...** aus dem Menü **Suchen** aus, geben Sie `POS1` im Textfeld ein und drücken Sie die EINGABETASTE.

POS1 wird im Vereinbarungs- und Definitionsabschnitt markiert, und neben dieser Bezeichnung ist die Ursache des Fehlers zu sehen. Die ENDE-TASTE wird durch die symbolische Konstante ENDET dargestellt, während der aus der On-Line-Hilfe kopierte **SELECT CASE**-Code die ENDE-TASTE durch ENDE-TASTE darstellt (siehe Abbildung 9.6).

Abbildung 9.6 Fehler in der Bezeichnung der symbolischen Konstante für die ENDE-TASTE



Es ist unwichtig, welche der beiden Bezeichnungen, ENDE-TASTE oder ENDT, beibehalten wird, solange das Wort in der Definition mit dem im Programm verwendeten Wort übereinstimmt. ENDE-TASTE läßt sich wie folgt in ENDET umwandeln:

28. Code-Eingabe in QKARTEN

1. Wählen Sie den Befehl **Ändern** aus dem Menü **Suchen** (ALT+S und D) aus. (Falls eine Meldung erscheint, daß das Programm neu gestartet werden muß, drücken Sie die EINGABETASTE.)
2. Geben Sie ENDE-TASTE im ersten Textfeld ein, drücken Sie die TAB-TASTE und geben Sie ENDET im zweiten Textfeld ein.
3. Beginnen Sie den Suchvorgang durch Drücken der EINGABETASTE. Der Suchbegriff kommt zum ersten Mal in der Prozedur RichtungsTaste vor.
4. Nehmen Sie die Änderung durch Drücken der Taste N vor. Der Suchvorgang wird beendet, da ENDE-TASTE nicht weiter vorkommt.

Schließen des Debug-Fensters

Bei geöffnetem Debug-Fenster wird das Programm langsamer ausgeführt als bei geschlossenem. Sie können das Debug-Fenster schließen, indem Sie zuerst den Befehl **Anzeigevariable löschen** aus dem Menü **Debug** wiederholt auswählen und dann die Variablen aus den entsprechenden Dialogfeldern markieren, bis das Debug-Fenster geschlossen wird. Es ist ebenfalls möglich, den Befehl **Alle Haltepunkte löschen** auszuwählen.

1. Wählen Sie den Befehl **Alle Haltepunkte löschen** aus dem Menü **Debug** durch Betätigen der Tastenkombination ALT+G und der Taste Z aus.
2. Wählen Sie den Befehl **Anzeigevariable löschen** aus dem Menü **Debug** durch Betätigen der Tastenkombination ALT+G und der Taste A aus.
3. Löschen Sie den Eintrag mit der EINGABETASTE.

Als Nächstes werden Sie das Programm wie folgt ausführen und speichern:

29. Code-Eingabe in QKARTEN

1. Wählen Sie den Befehl **Start** aus dem Menü **Ausführen** aus. QKARTEN müßte jetzt alle RICHTUNGSTASTEN und Tasten der Befehlszeilen erkennen und die entsprechende Aktion ausführen.
2. Wählen Sie **Beenden** aus der QKARTEN-Befehlszeile aus, und drücken Sie zum Fortfahren eine beliebige Taste.
3. Wählen Sie den Befehl **Speichern unter** aus dem Menü **Datei** aus, und speichern Sie das Programm unter dem Namen *endguelt.bas* ab.

   QKARTEN müßte jetzt voll einsatzfähig sein.

Automatische Prozedurvereinbarungen

Beim Speichern eines Programms erstellt QuickBASIC eine **DECLARE**-Anweisung für jede neue Prozedur. Solche **DECLARE**-Anweisungen sollten im Allgemeinen im Vereinbarungs- und Definitionsabschnitt des Programms eingesetzt werden. QuickBASIC überprüft Anzahl, Reihenfolge und Typ der beim Aufruf einer Prozedur übergebenen Argumente mit Hilfe der Prozedurvereinbarungen. **DECLARE**-Anweisungen sollten nach Anweisungen wie **DEF***typ* und **TYPE...END TYPE** am Ende der Vereinbarungen und Definitionen stehen. Kopieren Sie die `DECLARE`-Anweisung für die Prozedur `RichtungsTaste` in den für Prozedurvereinbarungen bestimmten Abschnitt (siehe auch die Vereinbarung für die Prozedur `Alarm` aus dem Kapitel 5):

30. Code-Eingabe in QKARTEN

1. Kehren Sie durch Drücken der Tastenkombination STRG+POS1 an den Anfang des Programms zurück.
2. Markieren Sie die gesamte Anweisung `DECLARE RichtungsTaste` durch Drücken der Tastenkombination UMSCHALT+ENDE.
3. Löschen Sie die ausgewählte Anweisung aus QKARTEN, und kopieren Sie diese in den Zwischenspeicher, indem Sie die Tastenkombination ALT+B und die Taste A drücken.
4. Gehen Sie zum Abschnitt der **SUB**-Prozedurvereinbarungen des Programms (ungefähr Zeile 80).
5. Erstellen Sie, falls erforderlich, eine Leerzeile, und fügen Sie die Anweisung `DECLARE RichtungsTaste` ein, indem Sie ALT+B und E betätigen.
6. Speichern Sie das Programm unter dem Namen *endguelt.bas*.

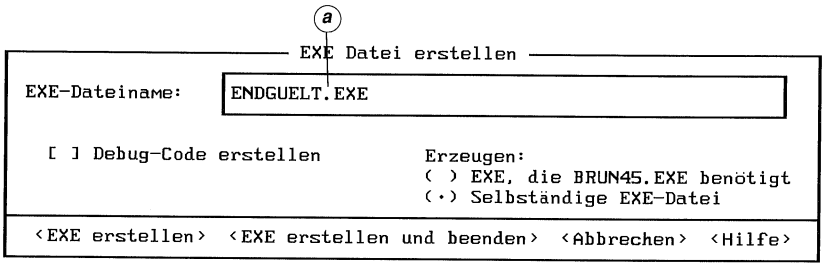
Erstellung einer selbständig ausführbaren Datei

Um eine direkt aus der DOS-Befehlszeile ausführbare Version von QKARTEN zu erstellen, ohne QuickBASIC zu starten, verfahren Sie wie folgt:

1. Wählen Sie den Befehl **EXE-Datei erstellen** aus dem Menü **Ausführen** (ALT+A und E).

   Falls ein Dialogfeld erscheint, das Sie zum Speichern der Dateien auffordert, drücken Sie die EINGABETASTE. Anschließend erscheint das Dialogfeld **EXE-Datei erstellen** (siehe Abbildung 9.7).

Abbildung 9.7 Das Dialogfeld EXE-Datei erstellen



a) Geben Sie den Namen der ausführbaren Datei an dieser Stelle ein.

2. Geben Sie `QKARTEN` im Textfeld ein.
3. Wählen Sie die Option **Selbtändige EXE-Datei** aus. (Es erscheint ein Punkt (•) neben der Option.)
4. Wählen Sie die Optionen **EXE erstellen und beenden** durch Betätigen der Tastenkombination ALT+U aus.

   QuickBASIC erstellt die EXE-Datei und verläßt DOS.
5. Führen Sie QKARTEN aus, indem Sie `qkarten` nach der DOS-Eingabeaufforderung eingeben und die EINGABETASTE drücken.

Hinweis Fall Sie nicht über eine Festplatte verfügen, fordert Sie QuickBASIC auf, eine Diskette mit der Datei *bc.exe* einzulegen und den zum Abschluß der ausführbaren Datei erforderlichen Pfad zu den Bibliotheken einzugeben. Die Dateien *bc.exe, link.exe* und *brun45.lib* befinden sich auf der 3. Diskette (Hilfsprogramme 1), während die Datei *bcom45.lib* auf der 4. Diskette (Hilfsprogramme 2) gespeichert ist. Geben Sie nach der entsprechenden Eingabeaufforderung einen vollständigen Pfad ein, der sowohl Laufwerksbezeichnung als auch Dateinamen enthält.

Kennenlernen der weiteren QuickBASIC-Menüpunkte

In den vorhergehenden Kapiteln wurden die meisten Befehle der QuickBASIC-Kurzmenüs vorgestellt, mit Ausnahme zweier Befehle. Der Befehl **Suchpfade festlegen** aus dem Menü **Optionen** erlaubt die Änderung einiger vom QuickBASIC-Installationsprogramm automatisch festgelegten Einstellungen. Der Befehl **Anzeigen Include-Datei** wird nur in Programmen verwendet, die auf Include-Dateien basieren. Vollständige Erläuterungen dieser Befehle sind Abschnitt 20.2, "Der Befehl Suchpfade festlegen", und 14.7, "Der Befehl Anzeigen Include-Datei", zu entnehmen.

Die Kurzmenüs bieten eine für den BASIC-Programmieranfänger oder Gelegenheitsprogrammierer besonders funktionelle Programmierumgebung. Das vollständige Menü von QuickBASIC verfügt über zusätzliche, für den fortgeschrittenen oder Berufsprogrammierer erforderliche Befehle. Das Menü **Debug** aus dem vollständigen Menü enthält beispielsweise die für das Debuggen umfangreicher und komplexer Programme notwendigen Befehle. Das Menü **Datei** aus dem vollständigen Menü ermöglicht das gleichzeitige Laden mehrerer Dateien, so daß Programme unter Verwendung mehrerer, separat kompilierter Quelldateien (oder Module) erstellt werden können. Die Kapitel 10 bis 21 aus diesem Handbuch befassen sich mit der Anwendung dieser Befehle. Die Programmiertechniken selbst werden im Handbuch *Programmieren in BASIC* behandelt.

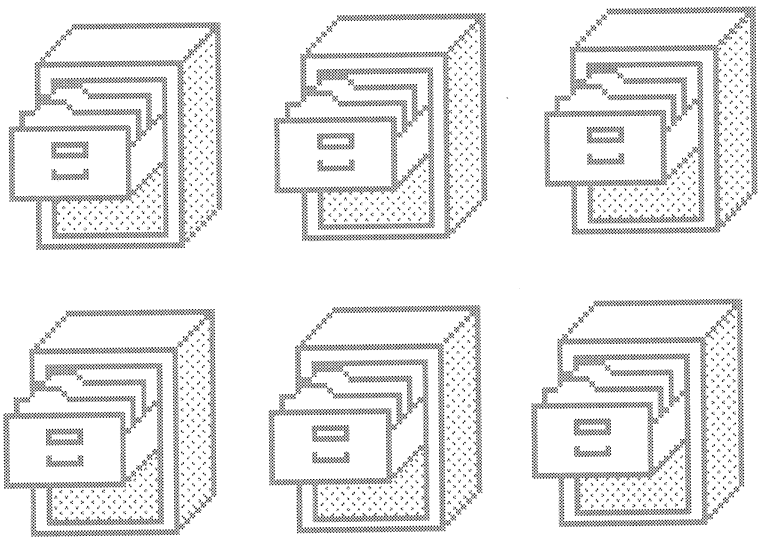
Weitere Informationen

Weitere Informationen zu den in diesen Kapiteln erläuterten Themen sind in nachstehenden Kapiteln zu finden:

Kapitel	***Themen***
Kapitel 2, "Prozeduren: Unterprogramme und Funktionen", aus dem Handbuch *Programmieren in BASIC*	Vollständige Informationen über das Programmieren mit Hilfe von Prozeduren, einschließlich der Regeln für Parameter und Argumente.
Kapitel 17, "Grundsätze und Techniken der Fehlerbeseitigung", Kapitel 18, "Das Menü Debug", und Kapitel 19, "Das Menü Aufrufe"	Vollständige Regeln für die Anwendung der QuickBASIC-Fehlerbeseitigung, einschließlich deren weiterer Hilfen, wie die Stoppbedingungen und das Menü Aufrufe.

3. Teil: QuickBASIC, Menüs und Befehle

3. Teil



QuickBASIC, Menüs und Befehle

Der 3. Teil, "QuickBASIC, Menüs und Befehle", bildet den Nachschlageteil zu den Microsoft QuickBASIC-Menüs und -Befehlen. Sie finden hier ausführliche Informationen, wie die Befehle funktionieren und in welchen Situationen sie benutzt werden.

Kapitel 10 und 11 führen die QuickBASIC-Umgebung ein und zeigen, wie man Dateien bearbeitet. Kapitel 12 bis 16 befassen sich mit den QuickBASIC-Einrichtungen, mit denen Programme erstellt, bearbeitet und ausgeführt werden können. In den Kapiteln 17 bis 19 werden sowohl allgemeine Techniken zur Fehlerbeseitigung als auch die Hilfen beschrieben, die ein schnelleres Debuggen ermöglichen. Kapitel 20 erläutert die Anpassung von QuickBASIC an die eigenen Bedürfnisse, während Kapitel 21 die Befehle zum Aufruf der On-Line-Hilfe erklärt.

Kapitelverzeichnis

10 Die Anwendung von QuickBASIC
11 Das Menü Datei
12 Die Verwendung des Editors
13 Das Menü Bearbeiten
14 Das Menü Ansicht
15 Das Menü Suchen
16 Das Menü Ausführen
17 Grundkonzepte und Techniken der Fehlerbeseitigung
18 Das Menü Debug
19 Das Menü Aufrufe (Vollständiges Menü)
20 Das Menü Optionen
21 Das Menü Hilfe

10 Die Anwendung von QuickBASIC

Dieses Kapitel führt in die QuickBASIC-Programmierumgebung ein und verdeutlicht deren Eigenschaften. Es enthält alle zur Anwendung der QuickBASIC-Programmierumgebung erforderlichen Informationen.

Es wird erklärt:

- Der Aufruf von QuickBASIC.
- Die Auswahl von Befehlen aus den Menüs.
- Die Auswahl von Elementen aus Dialogfeldern.
- Das Rollen von Text auf dem Bildschirm.
- Das Auswählen von Textpassagen in einem Dialogfeld oder einem Fenster.
- Das Ändern der Fenstergröße.
- Die Verwendung des Direktfensters.
- Die Verwendung des Debug-Fensters.
- Die Verwendung der Maus.
- Die Anwendung der On-Line-Hilfe.

10.1 Der Aufruf von QuickBASIC

Dieser Abschnitt erklärt den Aufruf von QuickBASIC mit und ohne Sonderoptionen und beschreibt die Bestandteile des QuickBASIC-Bildschirms. Bitte lesen Sie, falls erforderlich, Kapitel 1, "Installieren von Microsoft QuickBASIC", um QuickBASIC zu installieren.

10.1.1 Der QB-Befehl

Geben Sie nach der DOS-Eingabeaufforderung `QB` ein, um QuickBASIC zu starten.

Nachstehend steht die vollständige Syntax des `QB`-Befehls:

QB[[**/RUN**][*Programmname*][**/B**][**/G**][**/H**][**/NOHI**][**/C:***Puffergröße*]
[**/L**[*Bibliothekname*]][**MBF**][**/AH**][**/CMD** *Zeichenkette*]]

Hinweis In diesem Handbuch werden der QB-Befehl und dessen Optionen in Großbuchstaben geschrieben. Da DOS jedoch nicht zwischen Groß- und Kleinschreibung unterscheidet, können auch Kleinbuchstaben verwendet werden.

Nachstehende Liste führt die QuickBASIC-Befehlsoptionen auf. Diese Optionen können dem `QB`-Befehl folgend in der DOS-Befehlszeile eingegeben werden und wirken sich wie beschrieben aus. Die Optionen sind in der unten aufgeführten Reihenfolge einzusetzen.

Option	***Beschreibung***
/RUN *Programmname*	Veranlaßt QuickBASIC *Programmname* zu laden und sofort auszuführen.
Programmname	Benennt die beim Aufruf von QuickBASIC zu ladenden Dateien.
/B	Ermöglicht den Einsatz eines (schwarz-weiß) Farbmischmonitors mit einer Farbgraphikkarte. Mit dieser Option wird QuickBASIC auf einem Farbmonitor schwarz-weiß angezeigt.
/G	Weist QuickBASIC zur schnellstmöglichen Aktualisierung eines CGA-Bildschirms an. Die Option funktioniert nur bei Geräten, die über einen CGA-Monitor verfügen. Das Auftreten von "Bildflimmern" beim Aktualisieren des Bildschirms deutet darauf hin, daß die betreffende Hardware nicht in der Lage ist, die Option /G voll zu unterstützen. Rufen Sie QuickBASIC erneut ohne diese Option auf, um einen "klaren" Bildschirm zu erhalten.
/H	Die Anzeige erscheint in der höchsten von der Hardware unterstützten Auflösung.

Option	***Beschreibung***
/NOHI	Erlaubt den Einsatz eines Monitors, der keine intensive Anzeige unterstützt. Diese Option wird in Abschnitt 10.1.2 eingehend beschrieben.
/C:*Puffergröße*	Legt die Größe des Datenempfangspuffers fest. Diese Option funktioniert nur mit einer asynchronen Kommunikationskarte. Die Größe des Standard-Puffers beträgt 512 Byte, die Höchstgröße 32.767 Byte.
/L *Bibliotheksname*	Lädt die in *Bibliotheksname* angegebene Quick-Bibliothek. Falls kein Name angegeben wird, lädt QuickBASIC die Standard-Quick-Bibliothek, *qb.qlb*.
/MBF	Veranlaßt die Konversionsfunktionen von QuickBASIC, Zahlen im IEEE-Format als Zahlen im Microsoft Binärformat zu behandeln.
/AH	Mit Hilfe dieser Option können dynamische Felder, die aus Datensätzen, Zeichenketten fester Länge oder numerischen Daten bestehen, 64K überschreiten.
/CMD *Zeichenkette*	Übergibt *Zeichenkette* an die Funktion **COMMAND$** und muß als letzte Option in der Zeile stehen.

Geben Sie beispielsweise

```
QB /RUN GRAFIK /G /AH
```

ein, wenn Sie das Programm mit dem Namen `GRAFIK` mit folgenden Optionen ausführen möchten:

- Laden und sofortiges Ausführen von `Grafik`.
- Schnelles Aktualisieren des CGA-Bildschirms.
- Dynamische Datenfelder, deren Größe 64K überschreiten kann.

Im Allgemeinen wird QuickBASIC durch Eingabe von `QB` und Betätigen der EINGABETASTE gestartet.

10.1.2 Die Option /NOHI

Nach Eingabe von `QB` auf DOS-Ebene nimmt QuickBASIC an, daß der betreffende Monitor Farben intensiv (leuchtend) darstellen kann. Wenn das nicht der Fall ist, benötigt QuickBASIC Informationen zur Anzeige. Die zu diesem Zweck erforderlichen Optionen werden in nachstehender Liste aufgeführt:

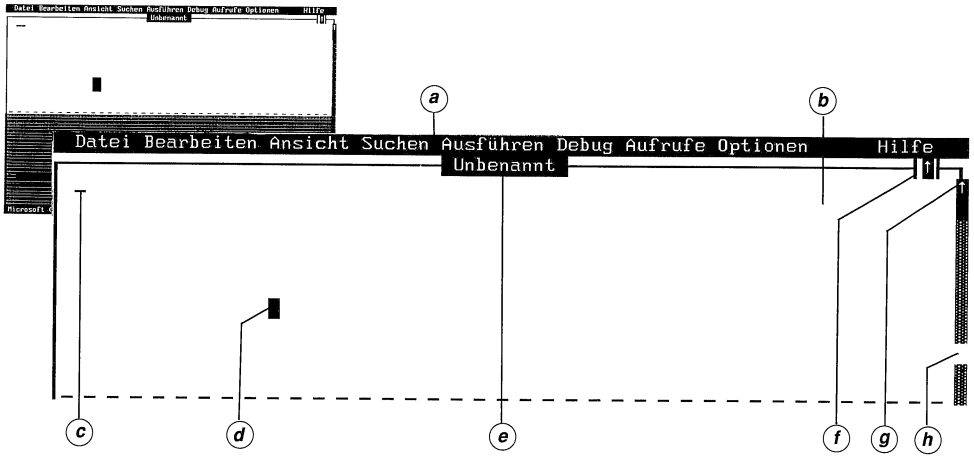
Monitoranzeige	***Aufrufbefehl***
16-farbig (CGA, EGA oder VGA)	`QB`
4-farbig, monochrom (MDA)	`QB`
8-farbig (CGA, EGA oder VGA)	`QB /NOHI`
4-farbige schwarz-weiß Farbmischanzeige (CGA, EGA oder VGA)	`QB /B`
2-farbige schwarz-weiß Farbmischanzeige (CGA, EGA oder VGA)	`QB /B /NOHI`

Laptop-Computer besitzen oft als 2-farbige Farbmischbildschirme eingestufte Flüssigkristalleinheiten (LCDs), die den Befehl `QB /B /NOHI` erfordern.

10.1.3 Die QuickBASIC-Anzeige

Wenn Sie QuickBASIC das erste mal aufrufen, erscheint ein Dialogfeld, das Ihnen die Möglichkeit bietet eine kurze Einführung zu erhalten. Sie können das Dialogfeld mit der ESC-TASTE löschen.Der QuickBASIC-Bildschirm ist in Abbildung 10.1 und 10.2 dargestellt. Abbildung 10.1 veranschaulicht die obere Hälfte des Bildschirms, die beim Aufruf von QuickBASIC erscheint.

Abbildung 10.1 Obere Hälfte des QuickBASIC-Bildschirms



a) Menüzeile.
b) Arbeitsfenster.
c) Cursor.
d) Mauscursor.
e) Titelleiste.
f) Vergrößerungsfeld.
g) Bildlaufpfeil.
h) Bildlauffeld.

Nachstehende Liste erläutert die in Abbildung 10.1 dargestellten Elemente:

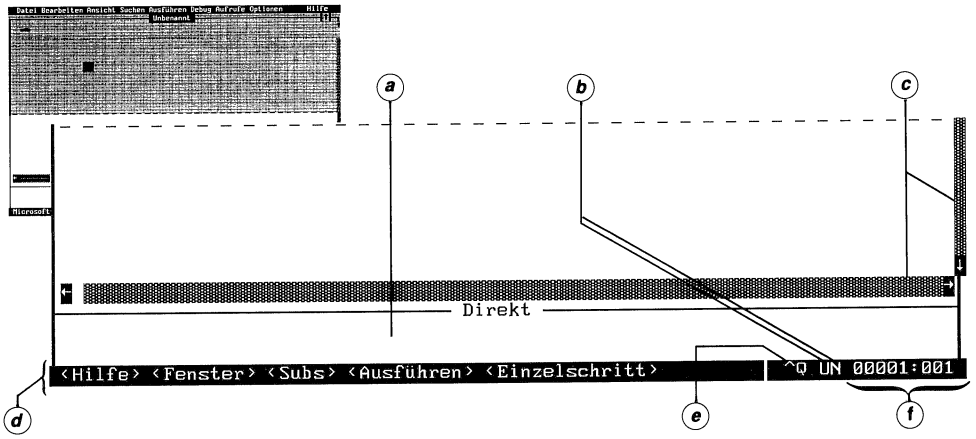
Element	***Beschreibung***
Menüzeile	Bezeichnet jedes Menü. Bei Drücken der ALT-TASTE zeigen die hervorgehobenen Buchstaben die jeweiligen Tasten an, die das betreffende Menü "öffnen".
Arbeitsfenster	Zeigt den Text des Programms an.
Cursor	Zeigt die Stelle, an der Text im jeweils aktiven Fenster eingegeben wird.
Mauscursor	Gibt die aktuelle Position der Maus auf dem Bildschirm an und kann nur mit der Maus verwendet werden.

Fortsetzung auf der folgenden Seite.

Element	*Beschreibung*
Titelleiste	Zeigt den Namen des Programms oder der aktuellen Prozedur an.
Vergrößerungsfeld	Vergrößert das aktive Fenster auf Gesamtbildschirmgröße und kann nur mit der Maus verwendet werden.
Bildlaufpfeil	Rollt den Text zeichen- oder zeilenweise und kann ebenfalls nur mit der Maus verwendet werden.
Bildlauffeld	Gibt die relative Cursorposition innerhalb der Datei oder der Prozedur an.

Abbildung 10.2 zeigt die untere Hälfte des Bildschirms nach Aufruf von QuickBASIC.

Abbildung 10.2 Untere Hälfte des QuickBASIC-Bildschirms



a) Direktfenster.
b) Anzeige der UMSCHALT-FESTSTELL- und NUM-FESTSTELLTASTE.
c) Bildlaufleisten.
d) Referenzzeile.
e) Befehlsanzeige im WordStar-Stil.
f) Zeilen- und Spaltenzähler.

Folgende Liste erläutert die in Abbildung 10.2 dargestellten Elemente:

Element	***Beschreibung***
Direktfenster	Führt QuickBASIC-Anweisungen direkt zeilenweise aus. Sie müßen nicht Teil eines Programms sein.
Anzeige UMSCHALT-FESTSTELL- und NUM-FESTSTELLTASTE	U weist darauf hin, daß die UMSCHALT-FESTSTELLTASTE, und N, daß die NUM-FESTSTELLTASTE eingeschaltet ist.
Bildlaufleisten	Rollen den Text im aktuellen aktiven Fenster und sind nur mit der Maus zu verwenden.
Referenzzeile	Enthält Referenzinformationen oder Befehlsflächen, die Befehlen aus den Menüs entsprechen. Das Anklicken eines Feldes wirkt sich wie das Ausführen des entsprechenden Befehles aus dem Menü aus. Falls Sie keine Maus besitzen, finden Sie am Ende dieses Handbuches eine Referenz mit den entsprechenden Tasten-kurzkombinationen.
Befehlsanzeige im WordStar-Stil	Bei Eingabe der Tastenkombination STRG+Q erscheint ^Q, ein Befehl im WordStar-Stil. ^K erscheint nach Festlegen eines Punktes zur Textmarkierung (siehe Abschnitt 12.4, "Verwendung von Punkten zur Textmarkierung"). Bei Eingabe eines literalen Zeichens (siehe Abschnitt 12.6, "Eingabe von Sonderzeichen") erscheint ^P.
Zeilen- und Spaltenzähler	Gibt die aktuelle Cursorposition im aktiven Fenster an.

Wenn Sie QuickBASIC ohne Angabe eines Programmnamens starten, können Sie mit dem Schreiben eines neuen Programms beginnen oder die On-Line-Hilfe anfordern. Ein allgemeines Hilfsfenster der On-Line-Hilfe zum jeweiligen Menü kann durch Drücken der Taste F1 angefordert werden. Ausführlichere Informationen sind im Menü **Hilfe** selbst zu finden (siehe Abschnitt 10.8, "Verwendung der On-Line-Hilfe", und Kapitel 21, "Das Menü Hilfe").

Ein Hilfsfenster läßt sich mit der ESC-TASTE löschen. Durch Betätigen der ESC-TASTE können auch Menüs, Dialogfelder und Fehlermeldungen gelöscht werden.

10.2 Öffnen der Menüs und Auswählen der Befehle

QuickBASIC-Befehle sind in Pulldown-Menüs auf der Menüzeile angeordnet. Abbildung 10.3 veranschaulicht eines dieser Menüs, das Menü **Datei**.

Abbildung 10.3 Das Menü Datei

Die QuickBASIC-Umgebung wurde für eine einfache und schnelle Anwendung entwickelt.Die meisten Befehle können unter Anwendung einer der folgenden Techniken durchgeführt werden:

- Öffnen von Menüs und Auswählen der Befehle über die Tastatur oder anhand der Maus. (Eine Zusammenfassung der Maus-Techniken ist in Abschnitt 10.7, "Verwendung der Maus", zu finden.)
- Unmittelbare Ausführung eines Befehls mit Hilfe einer "direkten Tastenkombination", die die Aufgabe eines Menübefehls übernimmt.

Drei nach einem Menübefehl stehende Punkte (...) deuten darauf hin, daß zur Ausführung des Befehls weitere Informationen erforderlich sind. In diesem Fall erscheint ein "Dialogfeld", das nach zusätzlichen Informationen fragt.

10.2.1 Anwendung der Tastatur

Jeder Befehl eines QuickBASIC-Menüs läßt sich über die Tastatur auswählen.

Ein Menü kann wie folgt geöffnet werden:

1. Drücken Sie die ALT-TASTE und lassen Sie diese wieder los. (Anschließend wird ein Buchstabe jedes Menüs hervorgehoben.)

2. Drücken Sie den hervorgehobenen Buchstaben der Menübezeichnung.

Hinweis Da die ALT-TASTE ein Umschalter ist, werden nach wiederholtem Drücken dieser Taste die Buchstaben wieder normal dargestellt. Die ALT-TASTE dient nur innerhalb der Menüs als Umschalter (drücken und loslassen). Beim Betätigen von Tastenkombinationen muß die erste betätigte Taste beim Drücken der restlichen Tasten niedergehalten werden.

Durch dieses Verfahren wird das gesamte Menü "ausgewählt" (hervorgehoben) und die Menübefehle sind auf dem Bildschirm zu sehen. Ein "aktiver" (verfügbarer) Befehl ist durch einen hervorgehobenen Buchstaben seines Namens gekennzeichnet.

Wenn kein Buchstabe einer Befehlsbezeichnung hervorgehoben ist, kann der entsprechende Befehl nicht durchgeführt werden. Wenn Sie beispielsweise während der Bearbeitung eines Programms keinen zum Kopieren oder Löschen bestimmten Text ausgewählt haben, sind die Befehle **Ausschneiden** und **Kopieren** nicht aktiv und können nur nach Markieren einer Textpassage angewandt werden. (In Abschnitt 12.2 wird die Textauswahl zum Kopieren oder Löschen beschrieben.)

Bewegen Sie sich wie folgt von einem Menü zum anderen:

- Schließen Sie das Menü mit der ESC-TASTE und wiederholen Sie die oben beschriebenen Arbeitsschritte 1 und 2.
- Drücken Sie die Richtungstasten NACH LINKS oder NACH RECHTS.

Wählen Sie einen Befehl durch eine der folgenden Methoden aus:

- Betätigen Sie die dem hervorgehobenen Buchstaben entsprechende Taste.
- Heben Sie den auszuführenden Befehl mit Hilfe der Richtungstasten NACH UNTEN oder NACH OBEN hervor und drücken Sie die EINGABETASTE.

Einige Befehle werden sofort ausgeführt und können nach Auswahl nicht mehr abgebrochen werden. Die von drei Punkten (...) gefolgten Befehle, dagegen, veranlassen die Anzeige eines Dialogfeldes, in dem zusätzliche Informationen eingegeben werden und die sich über die ESC-TASTE abbrechen lassen. Die Dialogfelder werden in Abschnitt 10.3 näher erläutert.

Siehe auch

Abschnitt 10.7, "Verwendung der Maus".

10.2.2 Einsatz der Tastenkombinationen zur direkten Befehlseingabe

In QuickBASIC dienen die Funktionstasten (F1 bis F10) als Tasten zur direkten Befehlseingabe für zahlreiche Menüfunktionen. Zahlreiche Tastenkombinationen erlauben Ihnen, die Auswahl eines Befehls aus den angezeigten Menüs durch ein oder zwei Tastenbetätigungen zu ersetzen. Die den jeweiligen Menübefehlen entsprechenden Tastenkombinationen werden im Menü neben den Befehlen aufgeführt. Als Programmieranfänger empfiehlt es sich, lediglich die angezeigten Menüs zu verwenden. Wenn Ihnen die QuickBASIC-Befehle geläufig sind, können Sie auch die Tastenkombinationen anwenden.

Um beispielsweise ein Programm auszuführen, kann sowohl der Befehl **Start** aus dem Menü **Ausführen** ausgewählt, als auch die direkte Tastenkombination UMSCHALT+F5 ohne Anwendung des Menüs betätigt werden. Tabelle 10.1 führt alle Tastenkombinationen zur direkten Befehlseingabe von QuickBASIC und die entsprechenden Menübefehle auf.

Tabelle 10.1 Die Tastenkombinationen in QuickBASIC

Tasten-kombination	***Funktion***	***Menübefehl***
UMSCHALT+F1	Zeigt eine allgemeine Hilfe an	Der Befehl **Überblick** aus dem Menü **Hilfe**
F1	Liefert Hilfsinformationen zu dem links vom Cursor stehenden Schlüsselwort oder der links vom Cursor stehenden Variablen	Der Befehl **Begriff** aus dem Menü **Hilfe**
ALT+F1	Zeigt bis zu 20 vorhergehende Hilfsanzeigen an	Keiner
F2	Zeigt eine Liste der geladenen **SUB**- oder **FUNCTION**-Prozeduren, Module, Include-Dateien oder Dokumentdateien an	Der Befehl **SUBs** aus dem Menü **Ansicht**
UMSCHALT+F2	Lädt die nächste Prozedur ins aktive Fenster	**Nächste SUB** aus dem Menü **Ansicht**
STRG+F2	Lädt vorhergehende Prozeduren ins aktive Fenster	Keiner

Tasten-kombination	*Funktion*	*Menübefehl*
F3	Sucht nach dem nächsten Vorkommen des zuvor angegebenen Textes	Der Befehl **Weitersuchen** aus dem Menü **Suchen**
F4	Schaltet den Ausgabe-bildschirm ein und aus	Der Befehl **Ausgabe-bildschirm** aus dem Menü **Ansicht**
F5	Setzt die Programmaus-führung ab der aktuellen Anweisung fort	Der Befehl **Weiter** aus dem Menü **Ausführen**
UMSCHALT+F5	Beginnt die Ausführung am Anfang des Programms	Der Befehl **Start** aus dem Menü **Ausführen**
F6	Aktiviert das nächste Fenster	Keiner
UMSCHALT+F6	Aktiviert das vorhergehende Fenster	Keiner
F7	Führt das Programm bis zur aktuellen Cursorposition aus	Keiner
F8	Führt die nächste Programmanweisung aus, Einzelschrittmodus	Keiner
UMSCHALT+F8	Geht die letzten 20, mit Hilfe des Befehls **Rückver-folgen ein** oder des Befehls **Verfolgen ein** erfaßten Programmanweisungen schrittweise rückwärts durch	Keiner
F9	Schaltet einen Haltepunkt ein und aus	Der Befehl **Haltepunkt ein/aus** aus dem Menü **Debug**
UMSCHALT+F9	Blendet das Dialogfeld **Aktuellen Wert anzeigen** ein	Der Befehl **Aktuellen Wert anzeigen** aus dem Menü **Debug**

Fortsetzung auf der folgenden Seite.

Tasten-kombination	*Funktion*	*Menübefehl*
F10	Führt die nächste Programmanweisung aus, Einzelschrittmodus außerhalb von Prozeduren	Keiner
UMSCHALT+F10	Geht die letzten 20, mit Hilfe des Befehls **Rückverfolgen ein** oder des Befehls **Verfolgen ein** erfaßten Programmanweisungen vorwärts durch	Keiner
STRG+F10	Schaltet zwischen mehreren Fenstern und dem aktiven Fenster in Gesamtbildschirmgröße um	Keiner

Siehe auch

Abschnitt 12.7, "Zusammenfassung der Editierbefehle"

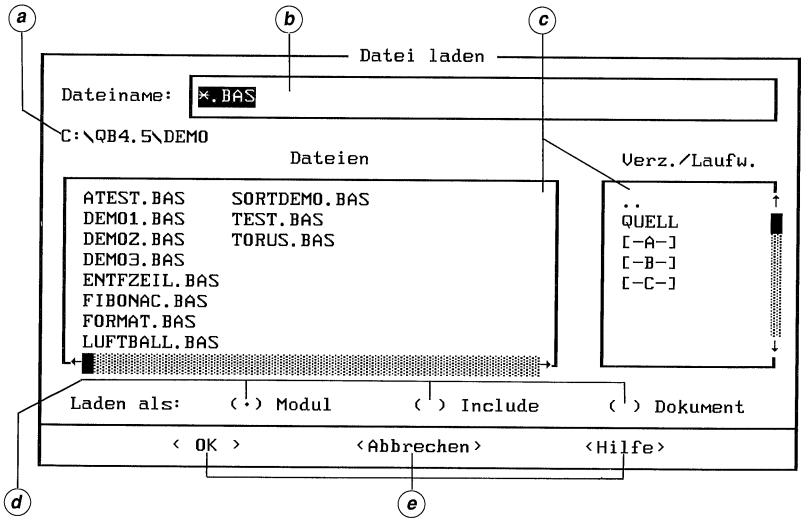
10.3 Verwendung der Dialogfelder

QuickBASIC zeigt ein Dialogfeld an, wenn es zur Durchführung eines Befehls zusätzliche Informationen benötigt. Ein Dialogfeld kann folgende Funktionen erfüllen:

- Den Namen einer Datei anfragen.
- Eine Optionsliste ausgeben.
- Das Überprüfen oder Abbrechen eines Befehls fordern.
- Den Programmierer auf einen Fehler aufmerksam machen.

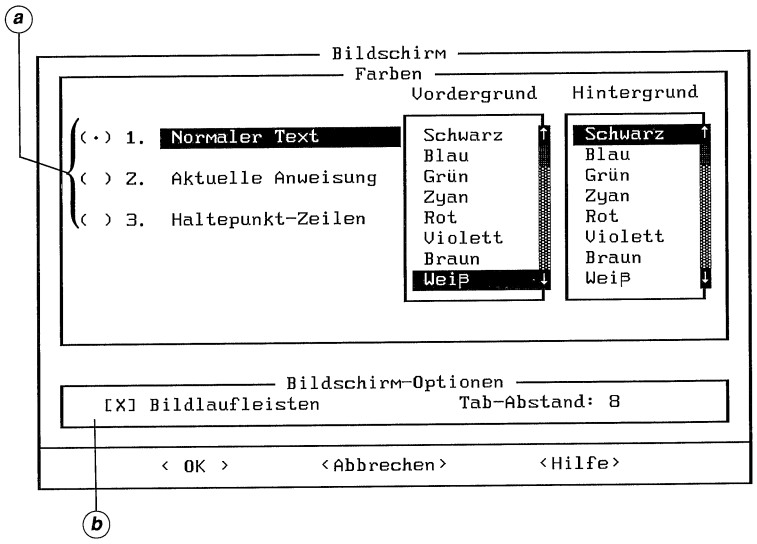
Abbildungen 10.4 und 10.5 veranschaulichen die Bestandteile eines Dialogfeldes.

Abbildung 10.4 Das Dialogfeld Lade Datei



a) Pfadangabe.
b) Textfeld.
c) Verzeichnisfelder.
d) Optionsflächen.
e) Befehlsflächen.

Abbildung 10.5 Das Dialogfeld Bildschirm



a) *Optionsflächen.*
b) *Kontrollfelder.*

Bewegen Sie sich mit Hilfe einer der folgenden Methoden von einem Teil des Dialogfeldes zum anderen:

- Drücken der TAB-TASTE.
- Betätigen der Taste, die dem hervorgehobenen Buchstaben aus dem gewünschten Teil entspricht, bei niedergedrückter ALT-TASTE. Wird ein Dialogfeld geöffnet, enthält es die zuletzt festgelegten Optionen.

Folgende Liste beschreibt die verschiedenen Bestandteile des Dialogfeldes:

Bestandteil	***Beschreibung***
Pfadangabe	Zeigt den Pfad zum aktuellen Verzeichnis an. Der Pfad läßt sich durch Eingabe eines neuen Pfadnamens im Textfeld oder durch Auswahl des gewünschten Verzeichnisses im Listenfeld **Verz./Laufw.** ändern.
Textfeld	Hier wird Text (z.B. ein Dateiname) eingegeben.
Listenfeld	Listet eine Gruppe ähnlicher Elemente auf (wie z.B. verfügbare Dateien oder Prozeduren), aus denen eines ausgewählt werden kann. Sie können sich mit Hilfe der RICHTUNGSTASTEN innerhalb der Listenfelder bewegen.

Bestandteil	*Beschreibung*
Kontrollfeld	Schaltet eine Option ein oder aus. Drücken Sie die LEERTASTE, wenn sich der Cursor im Bereich des Kontrollfeldes befindet, oder betätigen Sie die ALT+TASTE und dann den hervorgehobenen Buchstaben des ein- oder auszuschaltenden Elements. Ein X deutet auf eine eingeschaltete Option hin, während ein leeres Kontrollfeld bedeutet, daß die Option ausgeschaltet ist.
Optionsfläche	Wählt eine Option aus einer Optionsgruppe aus, wobei nicht mehr als eine Option gleichzeitig ausgewählt werden kann. Sie können sich mit Hilfe der RICHTUNGSTASTEN zwischen den Optionsflächen bewegen.
Befehlsfläche	Führt einen Befehl aus. Wenn sich der Cursor im Bereich der Befehlsflächen befindet, wählen Sie zwischen den Befehlsflächen mit der TAB-TASTE aus. Die Befehle werden durch Drücken der LEERTASTE oder der EINGABETASTE ausgeführt.

Es ist zu beachten, daß in Dialogfeldern die den hervorgehobenen Buchstaben entsprechenden Tasten bei niedergedrückter ALT-TASTE betätigt werden müssen. Dieses Verfahren unterscheidet sich von der Auswahl der Menübefehle, die das gleichzeitige Niederdrücken der ALT-TASTE und Betätigen des hervorgehobenen Buchstabens nicht erfordert.

10.4 Verwendung der Fenster

In Microsoft QuickBASIC werden vier verschiedene Fenster benutzt: Arbeits-, Direkt-, Hilfs- und Debug-Fenster. Dieser Abschnitt zeigt die Anwendung der QuickBASIC-Fenster:

- Wechseln des aktiven Fensters.
- Kopieren unterschiedlicher Programmteile in ein und aus einem aktiven Fenster.
- Ändern der Fenstergröße.
- Rollen des Textes innerhalb eines Fensters.

10.4.1 Die verschiedenen Fenster

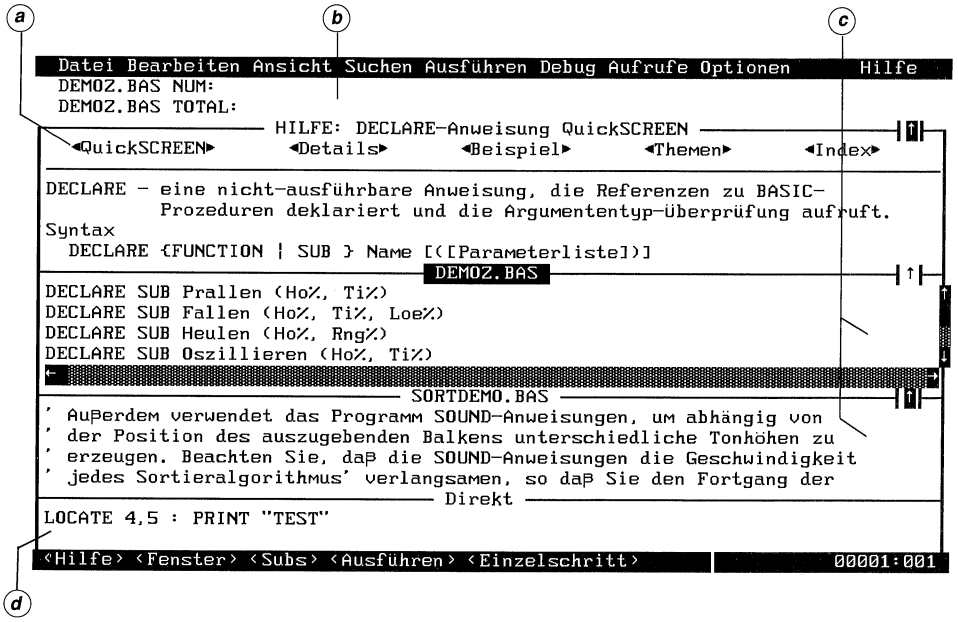
QuickBASIC verwendet verschiedene Fenster zur Durchführung unterschiedlicher Aufgaben. Nachstehende Liste beschreibt die in QuickBASIC verfügbaren Fenster:

Fenster	*Beschreibung*
Arbeitsfenster	Erscheint beim Aufruf von QuickBASIC als Hauptteil des Bildschirms (siehe Abbildung 10.1). Beim Laden eines Programms erscheint im Arbeitsfenster der als "Modul-Ebenen-Code" bezeichnete Programmtext. Das Menü **Bearbeiten** enthält Befehle, die es erlauben, unterschiedliche Programmteile auf einfache Weise in das und aus dem Arbeitsfenster zu laden. Um sich beispielsweise eine **FUNCTION**- oder **SUB**-Prozedur im Arbeitsfenster anzusehen, ist der Befehl **SUBs** aus dem Menü **Ansicht** und anschließend die entsprechende Prozedur aus dem Dialogfeld auszuwählen.
Direktfenster	Erscheint beim Aufruf von QuickBASIC am unteren Ende des Bildschirms und ermöglicht die unmittelbare Ausführung von BASIC-Anweisungen. (Nähere Informationen über das Direktfenster sind Abschnitt 17.3.5 zu entnehmen.)
Hilfsfenster	Enthält die On-Line-Hilfe. Aus dem Hilfsfenster können Beispiele kopiert und in das Programm eingefügt werden.
Debug-Fenster	Wird bei Auswahl bestimmter Befehle aus dem Debug-Menü am oberen Ende des Bildschirms geöffnet und zeigt Variablenwerte während des Programmablaufes an. (Weitere Informationen über die Verwendung des Debug-Fensters zur Fehlerbeseitigung sind in Abschnitt 17.3.4 zu finden.)

10.4.2 Teilen des Arbeitsfensters (nur vollständiges Menü)

Das Arbeitsfenster kann in ein oberes und ein unteres Fenster geteilt werden, um die gleichzeitige Bearbeitung von zwei unterschiedlichen Programmteilen zu ermöglichen. Dank dieser Eigenschaft (nur im vollständigen Menü verfügbar) ist es möglich, wie aus Abbildung 10.6 ersichtlich, gleichzeitig fünf Fenster auf dem Bildschirm zu öffnen: zwei Arbeitsfenster, das Direktfenster, das Hilfsfenster und das Debug-Fenster:

Abbildung 10.6 QuickBASIC-Anzeige mit fünf geöffneten Fenstern



a) Hilfsfenster.
b) Debug-Fenster.
c) Arbeitsfenster (zwei).
d) Direktfenster.

Wählen Sie den Befehl **Teilen** aus dem Menü **Ansicht**, um das Arbeitsfenster in zwei Teile zu teilen, und wiederholen Sie das Verfahren, um die ursprüngliche Anzeige wieder herzustellen.

Siehe auch

Abschnitt 14.3, "Der Befehl Teilen".

10.4.3 Wechseln des aktiven Fensters

Das Fenster, das den Cursor enthält, wird als "aktives" Fenster bezeichnet.

Aktivieren Sie ein anderes Fenster anhand folgender Arbeitsschritte:

- Bewegen Sie sich durch Drücken der Taste F6 fensterweise nach unten.
- Betätigen Sie die Tastenkombination UMSCHALT+F6, um von unten nach oben zu gehen.

10.4.4 Verändern der Fenstergröße

Die Fenstergröße kann zeilenweise vergrößert und verkleinert oder bis auf den Gesamtbildschirm erweitert werden. Um die Größe eines Fensters zu ändern, muß dieses zuerst aktiviert werden. Die Fenstergröße läßt sich dann mit Hilfe nachstehender Tastenkombinationen (wobei die erste Taste während der Betätigung der zweiten niedergehalten wird) unter Verwendung der PLUS- und MINUS-TASTE des Tastenblocks vergrößern oder verkleinern:

Tastenkombination	*Ergebnis*
ALT+PLUS(+)	Vergrößert das aktive Fenster um eine Zeile.
ALT+MINUS (-)	Verkleinert das aktive Fenster um eine Zeile.
STRG+F10	Erweitert das aktive Fenster auf Gesamtbildschirmgröße, oder stellt dessen ursprüngliche Dimensionen wieder her, falls das aktive Fenster diese Größe bereits erreicht hat. Wenn das Arbeitsfenster beim erstmaligen Betätigen der Tastenkombination STRG+F10 geteilt war, stellt diese Tastenkombination beide Fenster wieder her.

10.4.5 Rollen im aktiven Fenster

Um eine Datei bearbeiten zu können, die größer als das Arbeitsfenster ist, müssen Sie den Text nach unten und nach oben, so wie nach rechts und nach links bewegen ("rollen").

Drücken Sie die entsprechende RICHTUNGSTASTE, wenn eine Fensterkante erreicht wird, um mit dem Rollen des Textes zu beginnen. Um zum Beispiel zeichenweise nach rechts zu rollen, bringen Sie den Cursor zum äußerst rechten Zeichen neben der Bildschirmkante und drücken Sie die NACH RECHTS TASTE.

In Tabelle 10.2 finden Sie Informationen zum Rollen größerer Textpassagen. Die rechte Spalte dieser Tabelle zeigt die entsprechenden Tastenkombinationen im WordStar-Stil.

Tabelle 10.2 Rollen

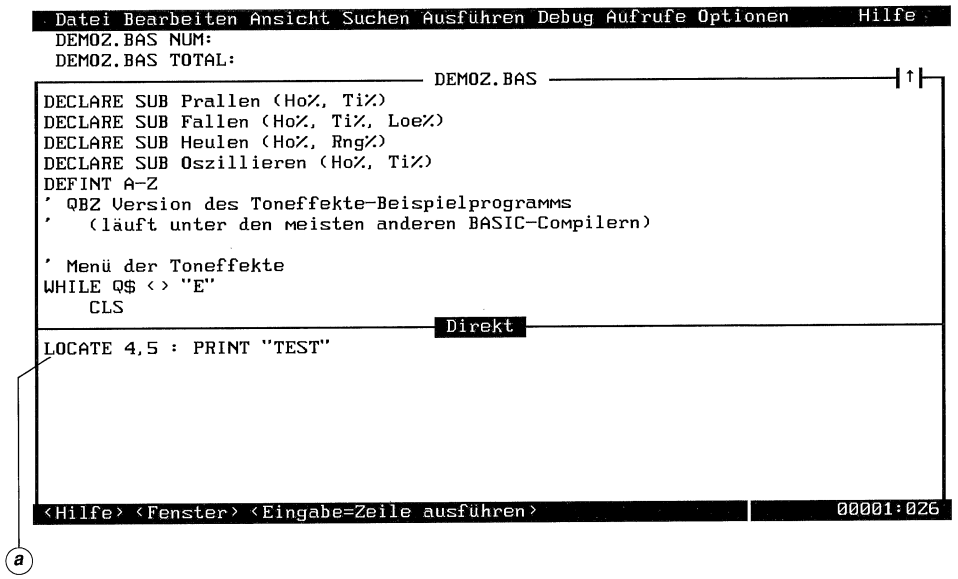
Rollaktion	***Tastenkombination***	***Tastenkombination im WordStar-Stil***
Zeilenanfang	POS1	STRG+Q+S
Zeilenende	ENDE-TASTE	STRG+Q+D
Seite nach oben	BILD NACH OBEN TASTE	STRG+R
Seite nach unten	BILD NACH UNTEN TASTE	STRG+C
Ein Fenster nach links	STRG+BILD NACH OBEN	--
Ein Fenster nach rechts	STRG+BILD NACH UNTEN	--
Dateianfang	STRG+POS1	STRG+Q+R
Dateiende	STRG+ENDE	STRG+Q+C

Es ist ebenfalls möglich, Markierungspunkte im Programmtext zu setzen und sich während der Bearbeitung von einem Markierungspunkt zum anderen zu bewegen. Nähere Informationen entnehmen Sie bitte Kapitel 12, "Die Verwendung des Editors".

10.5 Verwendung des Direktfensters

Das Direktfenster, das sich am unteren Ende des QuickBASIC-Bildschirms befindet, ermöglicht die unmittelbare Ausführung von QuickBASIC-Anweisungen. Mit Hilfe dieses Fensters können Teile des Programmcodes verbessert und deren Funktionsweise sofort getestet werden. Danach kann der Code in das Programm kopiert werden.

Abbildung 10.7 Das Direktfenster



a) *Zu testender Code.*

Im Direktfenster können bis zu zehn getrennte, auf 256 Zeichen beschränkte Zeilen eingegeben werden. Sie können sich auch im Direktfenster anhand der RICHTUNGSTASTEN bewegen. In jeder Zeile können mehrere Anweisungen vorkommen, unter der Bedingung, daß diese durch einen Doppelpunkt (:) voneinander getrennt werden. Wird die EINGABETASTE betätigt, werden jeweils die Anweisungen ausgeführt, in deren Zeile der Cursor steht.

Sie können beim Schreiben des Programms das Direktfenster aktivieren, um Anweisungen wie im Arbeitsfenster einzugeben, zu bearbeiten und sofort auszuführen. Somit können Programmierideen vor Eingliederung in ein umfangreicheres Programm getestet werden.

Das Direktfenster ähnelt dem Direktmodus von BASICA und weist folgende Merkmale auf:

- Beim Betätigen der EINGABETASTE in einer Zeile wird die zu dieser Zeile gehörende Anweisung bzw. Anweisungsgruppe ausgeführt.
- Im Direktfenster können bis zu zehn Zeilen eingegeben und in beliebiger Reihenfolge ausgeführt werden.

  Nach Eingabe der zehn Zeilen rollt jede neue Zeile den Text im Direktfenster um jeweils eine Zeile nach oben.
- Eine Zeile darf aus höchstens 256 Zeichen bestehen. Anweisungen können zwar in einer Zeile gruppiert werden, müssen jedoch durch Doppelpunkte (:) getrennt sein.
- Jede Zeile wird unabhängig von den anderen zum Direktfenster gehörenden Zeilen ausgeführt. Die in einer Zeile vorgenommenen Änderungen können sich jedoch auch auf andere Zeilen auswirken. Nachstehende Zeilen enthalten beispielsweise zwei Zuweisungen an die ganzzahlige Variable `x%`, so daß der in der nächsten Zeile ausgegebene Wert der zuletzt ausgeführten Zuweisungszeile entspricht:

  ```
  x% = 5
  print x%
  x% = 9
  ```
- Wenn das Direktfenster aktiv ist, läßt es sich durch wiederholtes Betätigen der Tastenkombination ALT+PLUS vergrößern und durch die Tastenkombination STRG+F10 auf den Gesamtbildschirm erweitern. Unabhängig von der Anzahl der angezeigten Zeilen sind jedoch nur die ersten zehn verwendbar. Die ursprüngliche Größe des Fensters wird durch Drücken der Tastenkombination STRG+F10 wieder hergestellt.
- Der im Direktfenster geschriebene Code läßt sich nicht mit Hilfe des Befehls **Speichern** auf Diskette/Festplatte abspeichern, sondern muß zuerst anhand der Befehle **Ausschneiden** und **Kopieren** aus dem Menü **Bearbeiten** in ein Arbeitsfenster kopiert und dann mit dem Rest der Datei gespeichert werden.

Siehe auch

Abschnitt 17.2, "Fehlervermeidung anhand von QuickBASIC", und Abschnitt 17.3.5, "Das Direktfenster".

10.5.1 Unzulässige Anweisungen

Die meisten QuickBASIC-Anweisungen und Funktionen sind auch im Direktfenster zulässig, mit Ausnahme der folgenden QuickBASIC-Schlüsselwörter, deren Eingabe eine Fehlermeldung im Direktfenster veranlassen:

COMMON	**ELSEIF**	**OPTION**
CONST	**END DEF**	**REDIM**
DATA	**END FUNCTION**	**SHARED**
DECLARE	**END IF**	**$STATIC**
DEF FN	**END SUB**	**STATIC**
DEF*typ*	**END TYPE**	**SUB**
DIM	**FUNCTION**	**TYPE**
DYNAMIC	**$INCLUDE**	**$DYNAMIC**

Hinweis Obwohl die Anweisung **END IF** im Direktfenster nicht zulässig ist, kann die einzeilige Form der Anweisung **IF...THEN...ELSE** verwendet werden.

10.5.2 Durchführung von Berechnungen

Im Direktfenster können komplizierte Ausdrücke mit Hilfe der BASIC-Funktionen (wie **SIN** oder **EXP**) berechnet und die Ergebnisse über die **PRINT**-Anweisung ausgegeben werden. Das Direktfenster kann ebenfalls zur Ausgabe des von **FRE(-1)** zurückgegebenen Wertes benutzt werden, der den nach Aufruf der QuickBASIC-Umgebung und Laden des Programms verbleibenden Speicherplatz anzeigt.

10.5.3 Testen der Bildschirmausgabe

Da die Größe und Komplexität eines zu entwickelnden Programms stetig zunimmt, ist es angebracht, Teilcode vor Eingliederung im Programm, im Direktfenster zu testen. Dadurch wird vermieden, daß nach jeder Änderung eines Codeteils das ganze Programm neu ausgeführt werden muß.

Beispiele

Sie können die Position der Bildschirmausgabe mit Hilfe folgender Zeilen im Direktfenster testen:

```
Zeil%=1 : Spal%=4*Zeil% : LOCATE Zeil%,Spal% : PRINT "."
```

Der Punkt aus der **PRINT**-Anweisung wird durch Ändern des Wertes der Variablen `Zeil%` und Drücken der EINGABETASTE an verschiedenen Stellen des Bildschirms ausgegeben.

Es kann vorkommen, daß ein Programm den Ausgabebildschirm nach Programmablauf in einem unerwünschten Modus zurückläßt (ein Graphikprogramm kann beispielsweise enden, ohne den Bildschirm auf den 80-spaltigen Textmodus zurückzusetzen). In diesem Fall kann der gewünschte Ausgabemodus mit Hilfe des Direktfensters festgelegt werden. Folgende Zeilen stellen zum Beispiel den 80-spaltigen Textmodus des Ausgabebildschirms wieder her und schließen jegliches im Programm geöffnetes Textdarstellungsfeld (ein beschränkter horizontaler Bildschirmbereich):

```
screen 0
width 80
view print
```

10.5.4 Prozeduraufruf

Die Auswirkungen einzelner **SUB**- oder **FUNCTION**-Prozeduren können durch deren Aufruf aus dem Direktfenster getestet werden. Führen Sie mit Hilfe des Direktfensters nur die zwischen den Anweisungen **SUB** und **END SUB** oder **FUNCTION** und **END FUNCTION** vorhandenen Anweisungen aus. Die Eingabe der folgenden Zeile in das Direktfenster veranlaßt beispielsweise die Ausführung sämtlicher Anweisungen der Prozedur `UnterProg1`, einschließlich der darin aufgeführten Aufrufe anderer Prozeduren:

```
call UnterProg1
```

10.5.5 Ändern der Variablenwerte

Anhand des Direktfensters können auch Variablen aus einem laufenden Programm neue Werte zugewiesen werden. Zu diesem Zweck ist das Programm anzuhalten, den Variablen ein neuer Wert im Direktfenster zuzuweisen und mit der Ausführung fortzufahren. Der im Direktfenster zugewiesene Wert wird von den Variablen bei Wiederaufnahme der Programmdurchführung angenommen.

Beispiel

Ein Variablenwert aus einem laufenden Programm läßt sich mit Hilfe folgender Arbeitsschritte im Direktfenster ändern:

1. Geben Sie dieses Programm im Arbeitsfenster ein:

```
FOR i% = 0 TO 10000
   LOCATE 10, 20 : PRINT i%
   LOCATE 10, 27 : PRINT " und zählt weiter"
NEXT i%
```

2. Starten Sie das Programm über den Befehl **Start** aus dem Menü **Ausführen** und unterbrechen Sie es durch Betätigen der Tastenkombination STRG+UNTBR.
3. Geben Sie

   `i% = 9900`

   im Direktfenster ein und drücken Sie die EINGABETASTE.
4. Setzen Sie das Programm durch Betätigen der Taste F5 fort.

   Nun wird die Schleife von 9.900 bis 10.000 ausgeführt.

Das Direktfenster kann nur auf eine Variable zugreifen, wenn dieser ein Wert zugewiesen wurde. Dazu muß zuerst eine Anweisung, die diese Variable enthält, im Arbeitsfenster ausgeführt werden. Wenn beispielsweise die zuletzt im Arbeitsfenster ausgeführte Anweisung der Variablen `x%` den Wert `3` zugewiesen hat, wird diese Zahl nach Eingabe von `PRINT x%` im Direktfenster und Drücken der EINGABETASTE auf dem Bildschirm ausgegeben.

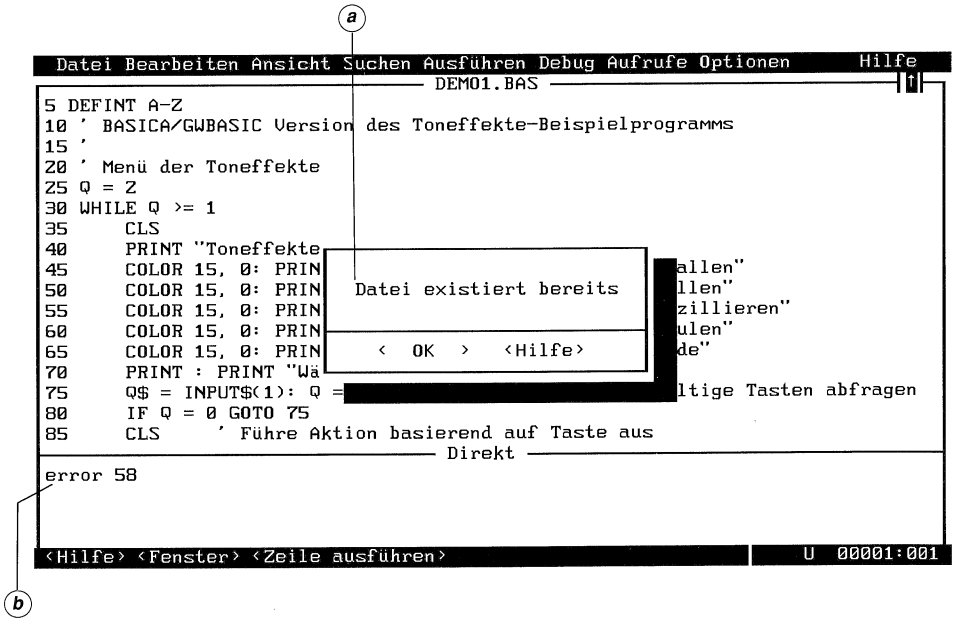
Wenn sich die nächste im Arbeitsfenster auszuführende Anweisung auf Modul-Ebene befindet, ist der Zugriff auf Variablen einer **SUB** oder **FUNCTION** aus dem Direktfenster nicht möglich.

10.5.6 Simulieren von Laufzeitfehlern

Laufzeitfehler treten während eines Programmablaufes auf und haben einen numerischen Code. Wenn das Programm eine Fehlerbehandlungsroutine verwendet (**ON ERROR**-Anweisungen), gibt QuickBASIC diesen numerischen Code an das Programm zurück.

Die dem numerischen Code entsprechende Fehlermeldung kann im Anhang I, "Fehlermeldungen", des Handbuches *Programmieren in BASIC* nachgelesen oder durch Simulieren des Fehlers herausgefunden werden. Ein Fehler läßt sich im Direktfenster durch Eingabe der betreffenden Zahl als Argument der **ERROR**-Anweisung simulieren. Anschließend zeigt QuickBASIC das Dialogfeld für den jeweiligen Fehler an, als ob der Fehler tatsächlich während eines Programmablaufes aufgetreten wäre. Abbildung 10.8 stellt ein für Fehlermeldungen bestimmtes Dialogfeld der QuickBASIC-Anzeige dar.

Abbildung 10.8 Simulierter Laufzeitfehler



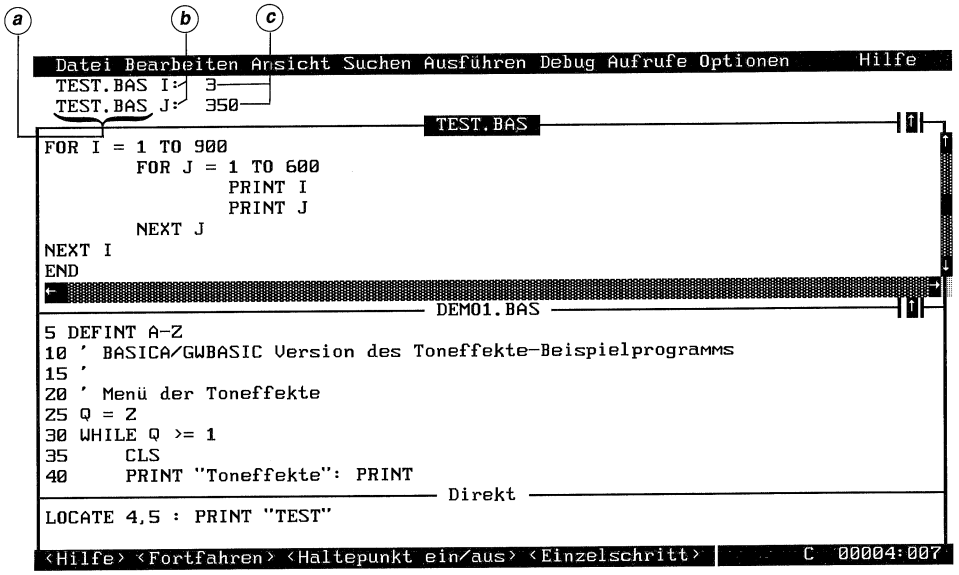
a) *Dialogfeld Fehlermeldung.*
b) *Im Direktfenster eingegebener Fehlercode.*

10.6 Verwendung des Debug-Fensters

Das Debug-Fenster zeigt beim Debuggen verfolgte Variablen an. Das Debug-Fenster wird am oberen Ende des Bildschirms geöffnet, wenn eine Stoppbedingung festgelegt oder eine Anzeigevariable hinzugefügt wird. (Siehe Abschnitt 17.3.2, "Haltepunkte und Stoppbedingungen", und Abschnitt 17.3.3, "Das Anzeigen von Variablen und Ausdrücken".)

Abbildung 10.9 zeigt den QuickBASIC-Bildschirm mit Einträgen im Debug-Fenster.

Abbildung 10.9 Das Debug-Fenster



a) *Programmstelle, an der die Variable hinzugefügt wurde.*
b) *Variablen (oder Ausdrücke).*
c) *Werte (oder Bedingungen).*

Das Debug-Fenster wird bei Ausführung des Befehls **Variable anzeigen**, des Befehls **Stoppbedingung** oder bei Auswahl des Elements **Anzeigen** aus dem Dialogfeld **Aktuellen Wert anzeigen** geöffnet. Die Befehle **Variable anzeigen**, **Stoppbedingung** und **Aktuellen Wert anzeigen** gehören alle zum Menü **Debug**. Jeder Eintrag im Debug-Fenster enthält folgende Informationen:

- Programmstelle, an der die Variable hinzugefügt wurde.

  Dieser Teil des Eintrages bezeichnet das Modul oder die Prozedur, aus der der Ausdruck im Debug-Fenster eingefügt wurde.

- Variable oder Ausdruck.

  Name der/des zu verfolgenden Variablen/Ausdrucks.

- Wert oder Bedingung.

  In Abbildung 10.9 haben die Variablen `I` und `J` die Werte `1` bzw. `12` angenommen. Da das Programm *test.bas* ausführt und sowohl `I` als auch `J` dem Debug-Fenster hinzugefügt wurden, während sich *test.bas* im Arbeitsfenster befand (worauf auch der vor `I` und `J` stehende Eintrag *test.bas* hindeutet), erscheinen die Werte beider Variablen. Wenn diese dagegen aus einer anderen Programmstelle, wie zum Beispiel einer Prozedur, hinzugefügt werden, erscheint während der Ausführung von *test.bas* im Debug-Fenster der Prozedurname und anstelle des Variablenwertes die Meldung `Nicht anzeigbar`.

Siehe auch

Abschnitt 17.3.4, "Das Debug-Fenster", und Kapitel 18, "Das Menü Debug".

10.7 Verwendung der Maus

QuickBASIC kann auch mit der Maus verwendet werden. Im folgenden werden die damit verbundenen Begriffe aus diesem Handbuch erläutert:

Mausbegriff	*Beschreibung*
Mauscursor	Blockcursor, dessen Bewegung auf dem Bildschirm der Bewegung der Maus entspricht.
Klicken	Positionieren des Mauscursors auf einem Element, Drücken und Loslassen der linken Maustaste.
Zweimal Klicken	Positionieren des Cursors auf einem Element und zweimaliges Drücken der linken Maustaste.
Ziehen	Positionieren des Mauscursors an den Anfang der auszuwählenden Textpassage, Drücken und Niederhalten der linken Maustaste. Bewegen des Mauscursors an das Ende der gewünschten Auswahl und Loslassen der Maustaste. Dieses Verfahren wird zum Hervorheben von Wörtern, Manipulieren der Fenstergrößen und Einsatz der Bildlaufleisten verwendet.

Befehle lassen sich mit Hilfe der Maus wie folgt auswählen:

1. Zeigen Sie auf die Menübezeichnung und klicken Sie die linke Maustaste.
2. Zeigen Sie auf den gewünschten Befehl und klicken Sie die linke Maustaste.

Hinweis QuickBASIC-Menüs und -Befehle lassen sich nur mit Hilfe der linken Maustaste auswählen. Das Klicken der rechten Maustaste wirkt sich wie das Betätigen der Taste F1 (Zugriff auf die On-Line-Hilfe) aus. Informationen, wie Sie die Funktion der rechten Maustaste ändern, um das Programm bis zu der Zeile mit dem Mauscursor auszuführen, entnehmen Sie bitte Abschnitt 20.3, "Der Befehl Rechte Maustaste".

Tabelle 10.3 beschreibt Maus-Techniken zur Durchführung anderer Aufgaben.

Tabelle 10.3 Mausbefehle

Aufgabe	*Maus-Technik*
Schließen eines Menüs	Mauscursor vom Menü entfernen und linke Maustaste klicken.
Aktivieren eines Fensters	Klicken innerhalb des Fensters.
Vergrößern oder verkleinern eines Arbeits- oder Direktfensters	Mauscursor auf der Titelleiste positionieren und Titelleiste nach oben bzw. nach unten ziehen.
Aktives Fenster auf Gesamtbildschirmgröße erweitern	Vergrößerungsfeld rechts von der Titelleiste anklicken oder Maus innerhalb der Titelleiste zweimal klicken.
Bildlauf	Cursor auf dem Bildlauffeld positionieren und zur Stelle der Leiste ziehen, die der gewünschten Position in der Datei entspricht. Um den Text seitenweise zu rollen, Mauscursor auf die Bildlaufleiste zwischen dem Bildlauffeld und dem oberen oder dem unteren Ende der Bildlaufleiste setzen und linke Maustaste klicken. Um zeilen- oder zeichenweise im Text zu rollen, Bildlaufpfeile an einem Ende der Bildlaufleiste anklicken.

Aufgabe	*Maus-Technik*
Auswählen von Programmen und Dateien	Dateinamen zweimal anklicken oder Dateinamen und Feld OK aus einem Listenfeld anklicken, (diese Technik ist mit den Befehlen **Programm laden**, **Zusammenführen** und **Lade Datei** anzuwenden), oder Dateiname durch einmaliges Klicken hervorheben und EINGABETASTE betätigen.
Verzeichnis wechseln	Verzeichnisnamen zweimal anklicken, um dessen Inhalt anzuzeigen. (In den meisten Dialogfeldern werden Verzeichnisse und Laufwerke auf der rechten Seite der Dateiliste separat aufgeführt). Wenn es sich beim aktuellen Verzeichnis um ein Unterverzeichnis handelt, erscheinen zwei Punkte (..) in der Liste, die zweimal angeklickt werden müssen, um die nächsthöhere Verzeichnisebene zu erreichen.
Dialogfeld **Aktuellen Wert anzeigen** auf dem Bilschirm erscheinen lassen	Rechte Maustaste bei niedergedrückter UMSCHALTTASTE klicken.

Hinweis Klicken Sie den Dateinamen aus dem Listenfeld zweimal an, um ein Programm mit der Maus zu laden. Das einmalige Anklicken eines Elements mit der Maus wählt das betreffende Element aus, während das zweimalige Anklicken eines Elements in einem Listenfeld das Element auswählt, sämtliche vorher getroffene Auswahlen bestätigt und das Dialogfeld schließt.

10.8 Verwendung der On-Line-Hilfe

QuickBASIC verfügt über ein umfassendes On-Line-Hilfssystem, das zu praktisch allen QuickBASIC-Begriffen, Menüs, Befehlen oder Schlüsselwörtern Informationen erteilt.

Siehe auch

Kapitel 21, "Das Menü Hilfe".

10.8.1 Hilfseinrichtungen

Die On-Line-Hilfe von QuickBASIC besteht aus zwei Teilen: der Umgebungshilfe (für Fehlermeldungen, Menüs und Befehle) und dem Microsoft QB-Ratgeber (für sprachorientierte Themen). Beide lassen sich mit der ESC-TASTE wieder vom Bildschirm löschen.

Die Umgebungshilfe erscheint in Dialogfeldern, der QB-Ratgeber im Hilfsfenster am oberen Ende des Bildschirms. Der QB-Ratgeber bietet folgendes:

- Hyperverbindungen, die zusammenhängende Themen verbinden und den Zugriff auf sämtliche mit einem bestimmten Begriff verknüpfte Informationen erlauben (siehe Abschnitt 10.8.2).
- Beispiele, die kopiert und direkt im Programm eingefügt werden können. Die Größe des Hilfsfensters kann verändert werden, es können Befehle wie BILD NACH OBEN oder die RICHTUNGSTASTEN benutzt werden. Der Text ist rollfähig. Es kann sogar der Befehl **Suchen...** aus dem Menü **Suchen** eingesetzt werden, um eine gewisse Information innerhalb einer bestimmten Hilfsanzeige zu finden.
- Punkte zur Textmarkierung sind ebenfalls zulässig. Es empfiehlt sich, in einer häufig verwendeten Anzeige einen Markierungspunkt wie in einem Programmtext einzusetzen, um die betreffende Anzeige zu einem späteren Zeitpunkt leicht wiederzufinden. (Siehe Abschnitt 12.4, "Verwendung von Punkten zur Textmarkierung".)
- QuickBASIC merkt sich die 20 zuletzt aufgerufenen Hyperverbindungen, da es einfacher ist, die Hilfsanzeigen durch Betätigen der Tastenkombination ALT+F1 zurückzuverfolgen, anstatt die On-Line-Hilfe wiederholt aufzurufen.

Es gibt vier Methoden des Zugriffs auf die On-Line-Hilfe:

1. Bringen Sie den Cursor zu dem Wort im Programm, für das Sie Hilfe benötigen, und drücken Sie die Taste F1.
2. Wählen Sie die Befehlsfläche **Hilfe** aus oder betätigen Sie die Taste F1, wenn ein Hilfsdialogfeld erscheint.
3. Fordern Sie weitere Informationen über zusammenhängende Begriffe anhand der Hyperverbindungen aus den Hilfsfenstern an (siehe Abschnitt 10.8.2).
4. Wählen Sie den entsprechenden Befehl aus dem Menü **Hilfe** aus.

10.8.2 Die Hyperverbindungen

Der QB-Ratgeber bietet als "Hyperbindungen" bezeichnete Informationen über zusammenhängende Themen. Der Zugriff auf alle Informationen zu einem bestimmten Thema erfordert nur wenige Tastenbetätigungen.

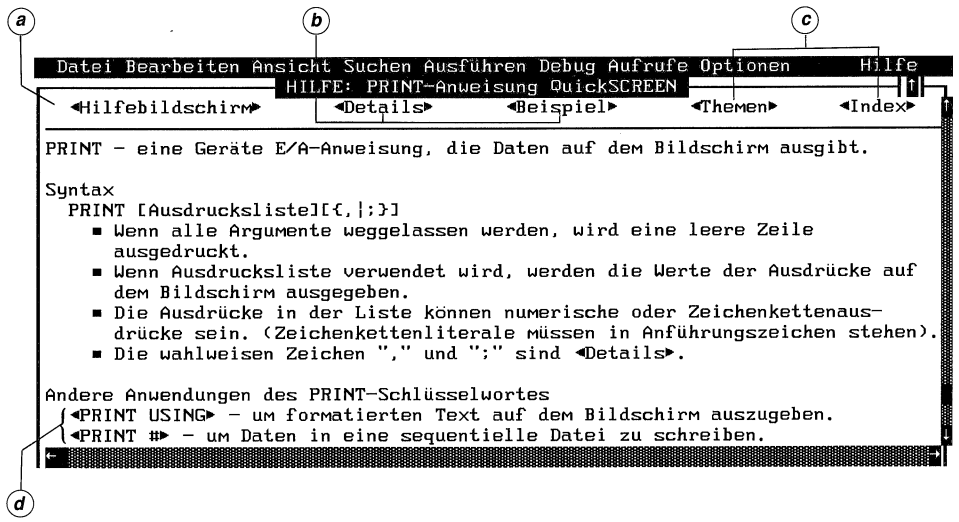
Die Hyperverbindungen erscheinen am oberen und unteren Ende des Hilfsfensters zwischen hervorgehobenen (bei monochromen Monitoren) oder grünen (bei den meisten Farbmonitoren) Dreiecken. Sämtliche BASIC-Schlüsselwörter im Programmtext bilden ihre eigene Hyperverbindung zur On-Line-Hilfe.

Aktivieren Sie das Hilfsfenster durch Betätigen der Tastenkombination UMSCHALT+F6, um Hyperverbindungen verwenden zu können, und bringen Sie den Cursor mit der TAB-TASTE auf die zu aktivierende Hyperverbindung. Sie haben auch die Möglichkeit, den ersten Buchstaben der Bezeichnung einer Hyperverbindung einzugeben, um den Cursor zu der betreffenden Hyperverbindung springen zu lassen. Die mit der jeweiligen Hyperverbindung verknüpfte Anzeige läßt sich durch Betätigen der Taste F1 aufrufen.

Hinweis Beim Verwenden der Maus kann der Mauscursor auf eine Hyperverbindung plaziert und die rechte Maustaste geklickt werden. Voraussetzung ist, daß diese Maustaste zum Aufrufen der kontextabhängigen Hilfe eingestellt ist (Standard-Einstellung).

Abbildung 10.10 veranschaulicht Beispiele von Hyperverbindungen in der **PRINT**-Hilfsanzeige. Die Hyperverbindungen `Details` und `Beispiel` enthalten zusätzliche Anmerkungen und Beispiele.

Abbildung 10.10 Die On-Line-Hilfe für die PRINT-Anweisung



a) Inaktive Hyperverbindungen (aktuelle Anzeige).

b) Hyperverbindungen zu zusätzlicher Hilfe für PRINT.

c) Hyperverbindungen zu allgemeinen Hilfsthemen.

d) Hyperverbindungen zu zusammenhängenden Anweisungen.

10.8.3 Bewegen innerhalb des Hilfsfensters

Innerhalb der Hilfsfenster können Sie sich mit der Tastatur, zum Beispiel den RICHTUNGSTASTEN oder der in Tabelle 10.4 beschriebenen Tastenkombinationen bewegen.

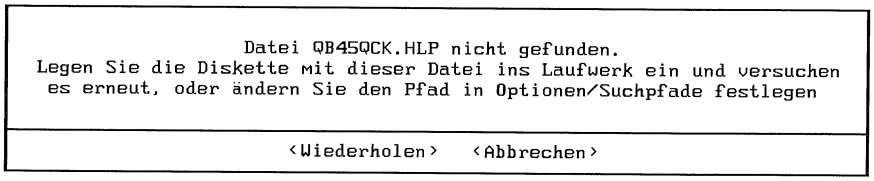
Tabelle 10.4 Die Hilfetasten

Tastenfolge	***Beschreibung***
UMSCHALT+F1	Zeigt **Überblick** an (ein Überblick zur Anwendung der On-Line-Hilfe).
ESC-TASTE	Schließt das Hilfsfenster.
TAB-TASTE	Geht zu der nächsten Hyperverbindung.
UMSCHALT+TAB	Geht zur vorhergehenden Hyperverbindung.
Zeichen	Geht zu einer nachfolgenden Hyperverbindung, die mit dem Buchstaben des *Zeichens* beginnt.
UMSCHALT+*Zeichen*	Geht zu einer vorhergehenden Hyperverbindung, die mit dem Buchstaben des *Zeichens* beginnt.
BILD NACH OBEN TASTE	Zeigt die vorhergehende Seite der Anzeige des aktuellen Begriffs an, falls eine solche besteht.
BILD NACH UNTEN TASTE	Zeigt die nachfolgende Seite der Anzeige des aktuellen Begriffs an, falls eine solche besteht.
STRG+F1	Zeigt Hilfe für den unmittelbar nach dem aktuellen Hilfsbegriff stehenden Begriff an.
UMSCHALT+STRG+F1	Zeigt Hilfe für den unmittelbar vor dem aktuellen Hilfsbegriff stehenden Begriff an.

10.8.4 Die Hilfsdateien

Beim Auswählen eines Hilfsbefehls sucht QuickBASIC nach der entsprechenden Hilfsdatei im laufenden Arbeitsverzeichnis sowie im Verzeichnis, das im Befehl **Suchpfade festlegen** aus dem Menü **Optionen** angegeben ist, und in jeder im System vorhandenen auswechselbaren Diskette/Platte. Wenn QuickBASIC die entsprechende Datei nicht finden kann, gibt es die in Abbildung 10.11 dargestellte Fehlermeldung aus. Kopieren Sie in diesem Fall die fehlende Datei in das entsprechende Verzeichnis .

Abbildung 10.11 Das Dialogfeld für die fehlende Hilfsdatei



Wenn kein Aufruf der On-Line-Hilfe möglich ist, kann es sein, daß sämtliche Hilfsdateien fehlen. In diesem Fall muß der Inhalt des QuickBASIC-Verzeichnisses überprüft oder die Installation wiederholt werden. Beim Versuch QuickBASIC gleichzeitig mit anderen speicherresidenten Dateien auszuführen, kann es vorkommen, daß der Speicherplatz für die Anzeige der On-Line-Hilfe nicht ausreicht. Löschen Sie in diesem Fall eines oder alle der anderen speicherresidenten Programme, und starten Sie QuickBASIC erneut. Der Speicherplatz ist ebenfalls ungenügend, wenn das System den in der Einleitung dieses Handbuches erwähnten Mindestanforderungen nicht entspricht.

10.8.5 Festplatten-System

Wenn Sie über eine Festplatte verfügen, wurden alle Hilfsdateien während der Installation in einem einzigen Verzeichnis angelegt (weitere Informationen finden Sie im 1. Kapitel, "Installieren von Microsoft QuickBASIC"). Der Befehl **Suchpfade festlegen** aus dem Menü **Optionen** gibt den Verzeichnispfad zu den Hilfsdateien an.

Beim Aufrufen sprachorientierter Hilfe erhalten Sie automatisch den QB-Ratgeber.

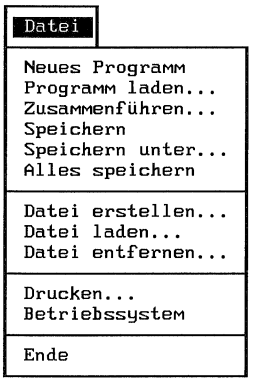
10.8.6 System mit Diskettenlaufwerken

Die Diskette, die QuickBASIC enthält, beinhaltet auch einen Teil der Sprachhilfe. Diese Datei umfaßt zwar Beschreibung und Syntax der Anweisungen, jedoch nicht die im QB-Ratgeber verfügbaren ausführlichen Beispiele.

Um den QB-Ratgeber in einem auswechselbaren Disketten/Platten-System zu verwenden, aktivieren Sie die Hyperverbindung `Details` oder `Beispiel` am oberen Ende des Bildschirms. Anschließend fordert Sie ein Dialogfeld auf, die Diskette mit dem Microsoft QB-Ratgeber einzulegen.

Nach Einlegen dieser Diskette können Sie auf ausführlichere Hilfe für die QuickBASIC-Schlüsselwörter zugreifen. Die Diskette kann im Laufwerk bleiben, bis Sie QuickBASIC zum Einlegen einer anderen Diskette auffordert.

11 Das Menü Datei



Mit dem Menü **Datei** lassen sich neue Dateien erstellen, vorhandene Dateien laden, Dateien verändern oder entfernen. Es können auch Dateien ausgedruckt und QuickBASIC beendet werden.

Dieses Menü verfügt über folgende Befehle:

- **Neues Programm**. Löscht alle zuvor geladenen Programme aus dem Arbeitsspeicher und ermöglicht das Schreiben eines neuen Programms.
- **Programm laden**. Lädt ein Programm; listet Dateien und Verzeichnisse auf.
- **Zusammenführen**. Führt die Inhalte zweier Dateien zusammen (nur im vollständigen Menü verfügbar).
- **Speichern**. Schreibt den Inhalt der Datei aus dem aktiven Arbeitsfenster in eine Diskettendatei (nur im vollständigen Menü verfügbar).
- **Speichern unter**. Speichert die aktuelle Datei unter dem angegebenen Namen.
- **Alles speichern**. Speichert sämtliche geladene Dateien (nur im vollständigen Menü verfügbar).
- **Datei erstellen**. Legt ein neues Modul, eine Include-Datei oder eine Dokumentdatei als Bestandteil des aktuellen Programms an (nur im vollständigen Menü verfügbar).

- **Datei laden.** Lädt eine vorhandene Datei, die aus einem Modul, einer Include-Datei oder einem Dokument besteht, in den Arbeitsspeicher (nur im vollständigen Menü verfügbar).
- **Datei entfernen.** Entfernt eine gesamte Datei aus dem Speicher (nur im vollständigen Menü).
- **Drucken.** Druckt das Programm ganz oder teilweise aus.
- **Betriebssystem.** Kehrt zeitweilig zur DOS-Befehlsebene zurück (nur im vollständigen Menü).
- **Ende.** Entfernt das Programm aus dem Arbeitsspeicher, beendet QuickBASIC und kehrt zur DOS-Befehlsebene zurück.

11.1 Der Befehl Neues Programm

Der Befehl **Neues Programm** aus dem Menü **Datei** löscht sämtliche zuvor geladenen Dateien, so daß Sie mit dem Schreiben eines neuen Programms beginnen können. Wenn der Text eines geladenen Programms nicht vollständig gespeichert wurde, fordert Sie QuickBASIC in einem Dialogfeld dazu auf.

Ein Programm enthält eine oder mehrere BASIC-Anweisungen, die von QuickBASIC in für Ihren Computer verständliche Anweisungen übersetzt werden. Ein BASIC-Programm kann aus einer einzigen Datei bestehen, QuickBASIC ermöglicht jedoch die Erstellung von Programmen, die aus mehreren selbständigen Teilen ("Modulen") bestehen. Normalerweise umfaßt jedes Modul Prozeduren zum Durchführen ganz bestimmter Aufgaben und kann von verschiedenen Programmen aufgerufen werden. Der Befehl **Neues Programm** erstellt das Hauptmodul eines mehrmoduligen Programms. In einem einmoduligen Programm bildet das Hauptmodul das einzige Modul.

Ein Programm kann vier Dateitypen aufweisen:

- Ein Hauptmodul (erforderlich)
- Andere Module
- Include-Dateien
- Dokumente (einschließlich .MAK-Dateien)

Module können folgende Elemente enthalten:

- Modul-Ebenen-Code
- **SUB**-Prozeduren
- **FUNCTION**-Prozeduren

Hinweis Der Befehl **Neues Programm** ist nur beim Schreiben eines vollkommen neuen Programms zu verwenden. Ein bereits bestehendes Programm ist mit Hilfe des Befehls **Programm laden** aufzurufen.

Siehe auch

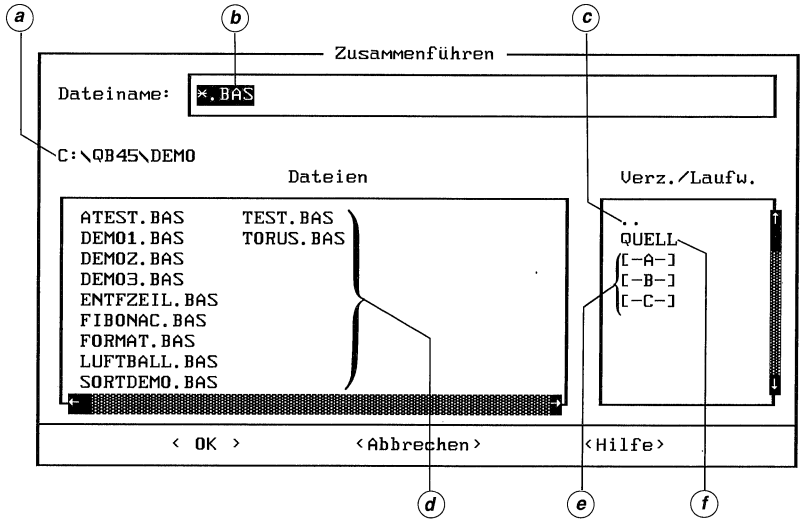
Abschnitt 11.2, "Der Befehl Programm laden", und Abschnitt 11.7, "Der Befehl Datei erstellen".

11.2 Der Befehl Programm laden

Mit Hilfe des Befehls **Programm laden** wird ein bestehendes Programm geladen. Das Dialogfeld führt die mit der Erweiterung *.bas* versehenen Dateien im aktuellen Arbeitsverzeichnis auf. Es ist auch möglich, Dateien in anderen Verzeichnissen oder auf anderen Disketten aufzulisten.

Bei Auswahl des Befehls **Programm laden** erscheint das in Abbildung 11.1 dargestellte Dialogfeld.

Abbildung 11.1 Das Dialogfeld Programm laden



a) Pfad der aktuell geladenen Datei.

*b) Die Eingabe bestimmt den angezeigten Dateityp (hier *.bas).*

c) Symbol des nächsthöheren Verzeichnisses.

d) .bas-Dateien im aktuellen Verzeichnis.

e) Laufwerke.

f) Unterverzeichnis.

Im Fall eines mehrmoduligen Programms werden alle zum Programm gehörenden Module geladen. Wenn ein Programm über Include-Dateien verfügt (siehe Abschnitt 2.5.4.3, "Verwendung von Include-Dateien für Deklarationen", und Anhang F, "Metabefehle", aus dem Handbuch *Programmieren in BASIC*), werden diese mitkompiliert, ohne geladen oder angezeigt zu werden.

11.2.1 Angabe einer Datei

Dateien werden im Verzeichnisfeld **Dateien** in mehreren Spalten angezeigt. Die Verzeichnisse und die verfügbaren Laufwerke erscheinen im Verzeichnisfeld **Verz./Laufw.** rechts vom Verzeichnisfeld **Dateien**. Die Inhalte beider Verzeichnisfelder lassen sich anhand der RICHTUNGSTASTEN rollen. Das Verzeichnisfeld **Dateien** rollt seitlich (nach links und nach rechts), während das Verzeichnisfeld **Verz./Laufw.** vertikal (nach oben und nach unten) rollt.

Im Dialogfeld **Programm laden** können die zu ladenden Dateien durch die zwei nachstehenden Methoden angegeben werden (Maus-Techniken sind Abschnitt 10.7, "Verwendung der Maus", zu entnehmen):

1. Geben Sie den Namen des Programms im Textfeld ein, und drücken Sie die EINGABETASTE.

   Wird ein Dateiname ohne Erweiterung eingegeben, nimmt QuickBASIC an, daß die Datei mit der Erweiterung *.bas* versehen ist. Soll tatsächlich eine Datei ohne Erweiterung geladen werden, ist unmittelbar nach dem Dateinamen ein Punkt (.) einzugeben.

2. Gehen Sie durch Betätigen der TAB-TASTE zum Verzeichnisfeld, bewegen Sie sich innerhalb des Verzeichnisfeldes mit Hilfe der RICHTUNGSTASTEN bis zur gewünschten Datei, und drücken Sie die EINGABETASTE.

   Sie können den Dateinamen ebenfalls hervorheben, indem Sie sich zum Verzeichnisfeld bewegen und die Taste für den ersten Buchstaben des gewünschten Dateinamens drücken.

In beiden Fällen erscheint der Programmtext im aktiven Fenster und der entsprechende Dateiname in der Titelleiste.

11.2.2 Auflisten des Verzeichnisinhalts

Anhand des Dialogfeldes **Programm laden** läßt sich der Inhalt jedes Verzeichnisses aufführen. Bei Auswahl eines Verzeichnisnamens im Verzeichnisfeld **Verz./Laufw.**, listet QuickBASIC alle zum ausgewählten Verzeichnis gehörenden Unterverzeichnisse und *.bas*-Dateien auf. Wird ein Dateiname im Verzeichnisfeld **Dateien** eingegeben oder aus diesem Feld ausgewählt, so wird die betreffende Datei im Arbeitsspeicher geladen. Folgende Liste beschreibt mehrere Methoden zum Auflisten des Verzeichnisinhalts:

Aufgabe	*Verfahren*
Auflisten aller Dateien im aktuellen Verzeichnis	Geben Sie *.* im Textfeld ein.
Aufführen der im Wurzelverzeichnis des Laufwerks A vorhandenen Dateien	Geben Sie A:/*.* im Textfeld ein.
Aufführen sämtlicher Dateien in einem Unterverzeichnis mit dem Namen BIN	Markieren Sie `BIN` im Verzeichnisfeld **Verz.Laufw.**, drücken Sie die EINGABETASTE, und geben Sie *.* im Textfeld ein.

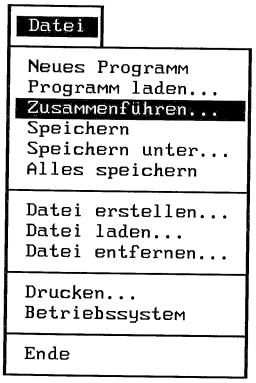
Aufgabe	*Verfahren*
Auflisten sämtlicher Dateien im aktuellen Verzeichnis mit der Erweiterung *.bi*	*.BI im Textfeld eingeben.
Aufführen sämtlicher, im Verzeichnis oberhalb des aktuellen Verzeichnisses vorhandenen Dateien	Geben Sie . . im Textfeld ein. Wenn Sie sich bereits im Verzeichnisfeld **Verz./Laufw.** befinden, drücken Sie eine RICHTUNGSTASTE oder den Punkt (.), um . . auszuwählen, und geben Sie *.* im Textfeld ein.

Hinweis Das "aktuelle Arbeitsverzeichnis" ist das vom Befehl **CD** aus DOS angezeigte Verzeichnis. Bei Auswahl eines Verzeichnisses aus einem Verzeichnisfeld werden jedoch die entsprechenden Dateien nur aufgelistet, ohne das aktuelle Arbeitsverzeichnis auf DOS-Ebene zu wechseln. Das aktuelle Verzeichnis kann durch Aktivieren des Direktfensters und Anwenden der **CHDIR**-Anweisung gewechselt werden.

Siehe auch

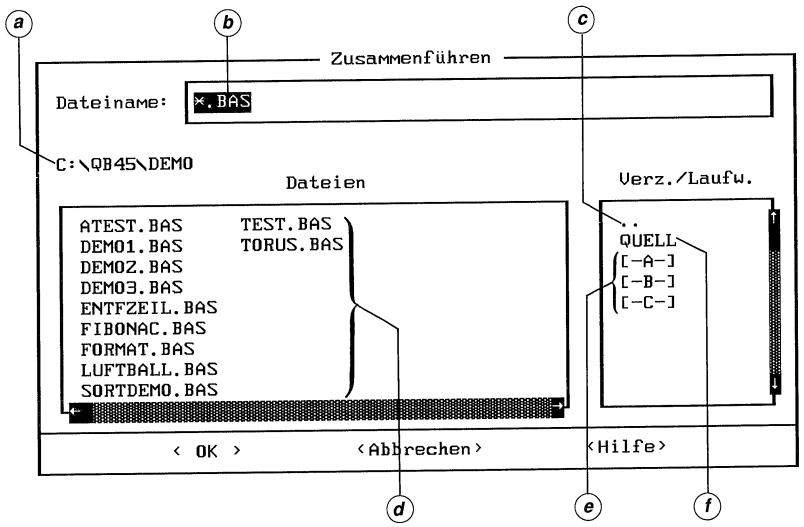
Abschnitt 11.1, "Der Befehl Neues Programm", und Abschnitt 11.8, "Der Befehl Datei laden".

11.3 Der Befehl Zusammenführen (nur im vollständigen Menü)



Mit Hilfe des Befehls **Zusammenführen** aus dem Menü **Datei** wird der Inhalt einer Datei am Ende der Zeile eingefügt, in der sich der Cursor befindet. Bei Auswahl des Befehls **Zusammenführen** aus dem Menü **Datei** erscheint das in Abbildung 11.2 dargestellte Dialogfeld.

Abbildung 11.2 Das Dialogfeld Zusammenführen



a) Pfad der aktuell geladenen Datei.
b) Die Eingabe bestimmt den angezeigten Dateityp.
c) Symbol des nächsthöheren Verzeichnisses.
d) .bas-Dateien im aktuellen Verzeichnis.
e) Laufwerke.
f) Unterverzeichnisse.

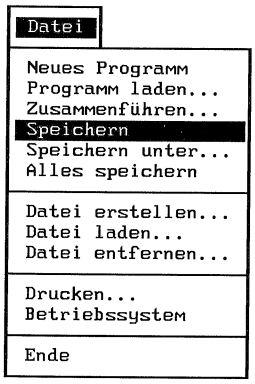
Das Dialogfeld **Zusammenführen** funktioniert wie das Dialogfeld **Programm laden**, jedoch mit unterschiedlichen Auswirkungen. Während der Befehl **Zusammenführen** eine gegebene Datei in die aktuelle Datei einfügt, löscht der Befehl **Programm laden** den aktuellen Inhalt des Arbeitsspeichers und lädt anschließend ein anderes Programm.

Hinweis QuickBASIC kann mit Hilfe des Befehls **Zusammenführen** keine Binärdateien, sondern nur Dateien im Textformat zusammenführen.

Siehe auch

Abschnitt 11.2, "Der Befehl Programm laden", und Abschnitt 14.1, "Der Befehl SUBs", dieses Handbuchs; Kapitel 7, "Programmieren mit Modulen", und Anhang B, "Unterschiede zu früheren QuickBASIC-Versionen", aus *Programmieren in BASIC*.

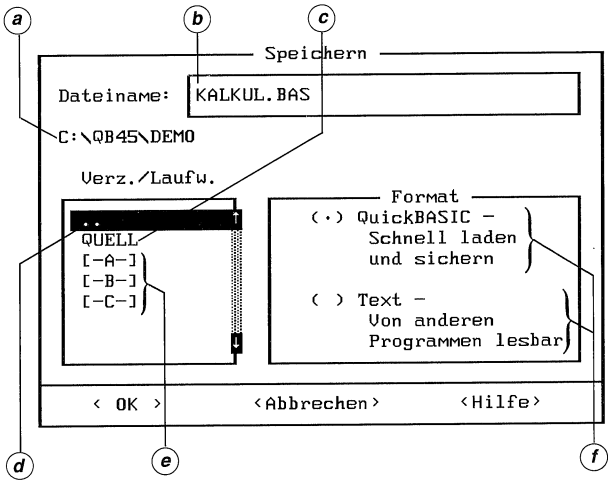
11.4 Der Befehl Speichern (nur im vollständigen Menü)



Der Befehl **Speichern** speichert den Inhalt der aktuellen, im aktiven Arbeitsfenster angezeigten Datei auf Diskette/Festplatte.

Wenn die zu speichernde Datei unter dem gleichen Namen bereits auf der Diskette vorhanden ist, wird diese Version der Datei durch den Befehl **Speichern** überschrieben. Wurde die Datei dagegen noch nicht benannt, erscheint das in Abbildung 11.3 dargestellte Dialogfeld **Speichern**, das Sie zur Eingabe eines Namens auffordert.

Abbildung 11.3 Das Dialogfeld Speichern



a) Pfad der aktuell geladenen Datei.
b) Dateiname.
c) Unterverzeichnis.
d) Symbol des nächsthöheren Verzeichnisses.
e) Laufwerke.
f) Optionen für das Datei-Format.

Siehe auch

Abschnitt 11.5, "Der Befehl Speichern unter", und Abschnitt 11.6, "Der Befehl Alles speichern".

11.5 Der Befehl Speichern unter

Der Befehl **Speichern unter** speichert die aktuelle Datei unter dem angegebenen Namen. Dieser Befehl wird zum Speichern einer Datei unter einem neuen Namen und zum Ändern des Datei-Formats eingesetzt. Beim Ändern des Dateinamens bleibt die alte Datei unter dem alten Namen weiter bestehen.

Bei Auswahl des Befehls **Speichern unter** aus dem Menü **Datei** erscheint ein Dialogfeld wie in Abbildung 11.3. Das Dialogfeld **Speichern unter** führt den Namen der zu speichernden Datei auf, der durch Eingabe eines neuen Namens im Textfeld geändert werden kann. Beim nächsten Speichern der Datei erscheint dieser neue Dateiname im Textfeld.

Nachstehende Liste beschreibt die im Dialogfeld **Speichern** vorhandenen Optionen für das Datei-Format:

Option	*Ziel*
QuickBASIC – Schnell laden und sichern	Speichert ein Programm im QuickBASIC-Format. Ein auf diese Weise gespeichertes Programm läßt sich zwar schneller als eine im Textformat gespeicherte Datei laden, kann jedoch nur unter Anwendung von QuickBASIC bearbeitet werden, da es sich in diesem Fall um keine ASCII-Datei handelt. Dieses Format wird beim Speichern eines Programms standardmäßig verwendet. Dateien, die anhand der Befehle **Datei erstellen** oder **Datei laden** als Dokumente oder Include-Dateien klassifiziert wurden, können nicht in diesem Format gespeichert werden.
Text – Von anderen Programmen lesbar	Die Datei wird als ASCII-Textdatei gespeichert. Die Textdateien können mit Hilfe jedes Editors oder Textverarbeitungssystems, der/das ASCII-Dateien erkennt, gelesen, verändert oder ausgegeben werden. Als Dokumente oder Include-Dateien klassifizierte Dateien werden ausschließlich in diesem Format gespeichert.

Siehe auch

Abschnitt 11.4, "Der Befehl Speichern", und Abschnitt 11.6, "Der Befehl Alles Speichern".

11.6 Der Befehl Alles Speichern (nur im vollständigen Menü)

Der Befehl **Alles Speichern** speichert sämtliche aktuell geladenen Dateien, die nach dem letzten Speichervorgang geändert wurden. Eine aktuell geladene Datei ist eine Datei, deren Name im Dialogfeld des Befehls **SUBs** aus dem Menü **Ansicht** erscheint. Der Befehl **Alles Speichern** ist bei der Arbeit mit mehreren Dateien im Arbeitspeicher nützlich.

Wenn das zu speichernde Hauptmodul bereits einen Namen hat, überschreibt der Befehl **Alles Speichern** die bestehende Version. Andernfalls erscheint ein Dialogfeld wie in Abbildung 11.3, das Sie zur Eingabe eines Namens auffordert. Im Abschnitt 11.5 werden die Optionen des Datei-Formats im einzelnen beschrieben.

Beim Speichern eines mehrmoduligen Programms erstellt QuickBASIC eine spezielle Datei, die alle im Programm vorhandenen Module aufführt. Die Datei wird mit dem Basisnamen des Hauptmoduls und der Erweiterung *.mak* bezeichnet. Wenn das Programm erneut geladen wird, verwendet QuickBASIC die MAK-Datei als "Wegweiser" zu den verschiedenen Modulen des Programms. Nähere Informationen zu den MAK-Dateien sind Abschnitt 16.7.2 zu entnehmen.

Siehe auch

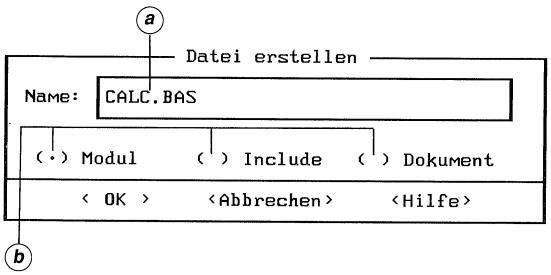
Abschnitt 11.4, "Der Befehl Speichern", und Abschnitt 14.1, "Der Befehl SUBs".

11.7 Der Befehl Datei erstellen (nur im vollständigen Menü)

Der Befehl **Datei erstellen** aus dem Menü **Datei** ermöglicht das Erstellen einer neuen Datei, als Teil des aktuellen Programms.

Bei Auswahl des Befehls **Datei erstellen** aus dem Menü **Datei** erscheint ein Dialogfeld. Eine neue Datei läßt sich dann durch Eingabe des Dateinamens im Textfeld, Auswählen des Dateityps und Betätigen der EINGABETASTE erstellen (siehe Abbildung 11.4).

Abbildung 11.4 Das Dialogfeld Datei erstellen



a) Dateiname.

b) Optionen für den Dateityp.

Wählen Sie den geeigneten Dateityp aus folgender Liste aus:

Dateityp	*Beschreibung*
Modul	Bildet einen diskreten Programmbestandteil. Ein Programm besteht aus mindestens einem Modul. Beim Erstellen eines mehrmoduligen Programms, wird jedes Modul als separate Datei des Programms auf Diskette/Platte abgespeichert. Module erweisen sich zweckmäßig in langen Programmen oder bei wiederholter Anwendung derselben Prozedurengruppe in verschiedenen Programmen. Der intelligente Editor von QuickBASIC überprüft Syntax und Formatierung der Module. Weitere Informationen zum intelligenten Editor sind in Abschnitt 12.5 dieses Handbuches zu finden. Nähere Informationen über Module entnehmen Sie bitte Kapitel 7, aus *Programmieren in BASIC*.
Include	Bezeichnet eine Textdatei, deren Anweisungen im Programm mitkompiliert werden, wenn QuickBASIC den betreffenden Dateinamen nach einem Metabefehl **$INCLUDE** antrifft. Der intelligente Editor von QuickBASIC überprüft Syntax und Formatierung von Include-Dateien sowie von Modulen. Der Inhalt der Include-Datei wird im Programm verwendet, obwohl die Include-Datei selbst nicht zum Programm gehört. Der Name dieser Datei wird in der MAK-Datei eines mehrmoduligen Programms nicht aufgeführt.
Dokument	Bezeichnet eine Textdatei. Eine als Dokument geöffnete Datei wird von QuickBASIC nicht als BASIC-Code behandelt und deren Syntax und Formatierung werden somit nicht vom intelligenten Editor überprüft. Ein Dokument kann jedoch mit dem QuickBASIC-Editor genau wie mit einem normalen Textverarbeitungssystem bearbeitet werden. Die Namen von Dokumenten erscheinen niemals in einer MAK-Datei.

Nach Erstellung eines der oben aufgeführten Dateitypen in QuickBASIC erscheint der Dateiname im Verzeichnisfeld des Dialogfeldes **SUBs** aus dem Menü **Ansicht**.

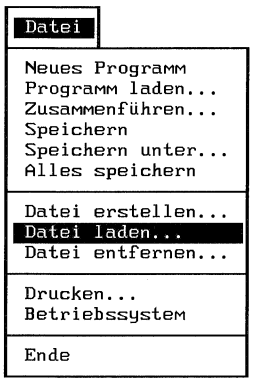
Solange der Name einer Datei im Verzeichnisfeld erscheint, kann diese unter keinen Umständen, auch nicht unter einem anderen Dateityp, erneut geladen werden. Wenn beispielsweise eine Datei als Dokument erstellt wird, kann diese nicht erneut als Include-Datei geladen werden, ohne zuvor die aktuelle Version anhand des Befehls **Datei entfernen** aus dem Arbeitsspeicher zu löschen (siehe Abschnitt 11.9, "Der Befehl Datei entfernen").

Hinweis Benutzen Sie den Befehl **Datei erstellen**, um dem aktuellen Programm ein neues Modul hinzuzufügen. Dies unterscheidet den Befehl **Datei erstellen** vom Befehl **Programm laden**, der eine Datei als Modul lädt, nachdem alle zuvor in den Arbeitsspeicher geladenen Dateien entfernt wurden. Wenn beim Anlegen eines Moduls mit Hilfe des Befehls **Datei erstellen** keine weiteren Module geladen sind, wird dieses als Hauptmodul des Programms betrachtet.

Siehe auch

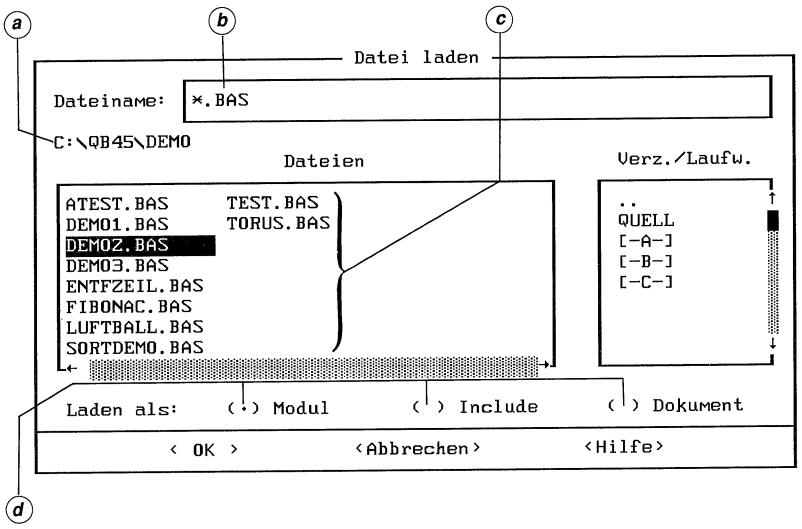
Abschnitt 11.1, "Der Befehl Neues Programm", Abschnitt 11.2, "Der Befehl Programm laden", Abschnitt 11.8, "Der Befehl Datei laden", und Abschnitt 14.1, "Der Befehl SUBs".

11.8 Der Befehl Datei laden (nur im vollständigen Menü)



Der Befehl **Datei laden** aus dem Menü **Datei** lädt einzelne Module des Programms, sowie Include-Dateien und Dokumente in den Arbeitsspeicher. Bei Auswahl des Befehls **Datei laden** erscheint ein Dialogfeld. Geben Sie den Dateinamen im Textfeld ein, wählen Sie den Dateityp aus und drücken Sie die EINGABETASTE, um eine Datei zu laden (siehe Abbildung 11.5).

Abbildung 11.5 Das Dialogfeld Datei laden



a) Pfad der aktuell geladenen Datei.

b) Zu ladende Datei.

c) .bas-Dateien im aktuellen Verzeichnis.

d) Optionen für den Dateityp.

Als erstes muß der korrekte Dateityp, sei es Modul, Include-Datei oder Dokument, ausgewählt werden. Weitere Einzelheiten über jeden Dateityp finden Sie in Abschnitt 11.7, "Der Befehl Datei erstellen".

Nach Laden eines Dateityps in QuickBASIC, erscheint der Dateiname im Verzeichnisfeld des Dialogfeldes **SUBs** aus dem Menü **Ansicht**.

Während eine Datei in diesem Verzeichnisfeld erscheint, kann diese unter keinen Umständen, auch nicht unter einem anderen Dateityp, erneut geladen werden. Wird beispielsweise eine Datei als Dokument geladen, kann diese nicht erneut als Include-Datei geladen werden, ohne sie zuvor anhand des Befehls **Datei entfernen** aus dem Arbeitsspeicher zu löschen. Siehe Abschnitt 11.9 für weitere Informationen zum Befehl **Datei entfernen**.

Falls es in anderen Modulen (Programmdateien) Prozeduren gibt, die Sie im aktuellen Programm benutzen möchten, können Sie diese Module mit dem Befehl **Datei laden** laden. Anschließend kann auf den darin enthaltenen Prozeduren beliebig oft im Programm Bezug genommen werden.

Das aktuell im Arbeitspeicher vorhandene Programm kann wie folgt mit Modulen aus anderen Programmen ergänzt werden.

1. Wählen Sie den Befehl **Datei laden** aus dem Menü **Datei**.
2. Wählen Sie den Namen des Moduls aus, der die im aktuellen Programm einzusetzenden Prozeduren enthält.
3. Laden Sie eine Datei als Modul durch Betätigen der EINGABETASTE. Die Option **Modul** ist bereits ausgewählt.
4. Wiederholen Sie die Arbeitsschritte 1 bis 3 für jedes Modul, das geladen werden soll.

Nach Laden aller gewünschten Module speichern Sie das Programm mit Hilfe des Befehls **Alles Speichern**.

Siehe auch

Abschnitt 11.1, "Der Befehl Neues Programm", und Abschnitt 11.7, "Der Befehl Datei erstellen".

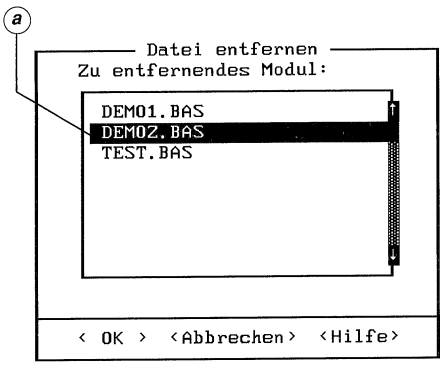
11.9 Der Befehl Datei entfernen (nur im vollständigen Menü)

Der Befehl **Datei entfernen** löscht eine Include-Datei, ein Dokument oder ein gesamtes Modul aus dem Arbeitsspeicher. Die entsprechende Diskettendatei bleibt zwar weiter bestehen, beim Speichern wird das Modul jedoch aus dem aktuellen Programm entfernt.

Wenn das Programm aus mehreren Modulen besteht, wird der Name des entfernten Moduls aus der MAK-Datei des Programms gelöscht. Wird das Hauptmodul eines mehrmoduligen Programms entfernt, fordert Sie QuickBASIC auf, zunächst ein neues Hauptmodul festzulegen. Nach Entfernen eines Dokuments oder einer Include-Datei werden die restlichen Dateien unverändert im Arbeitsspeicher beibehalten.

Bei Auswahl des Befehls **Datei entfernen**, erscheint das in Abbildung 11.6 gezeigte Dialogfeld.

Abbildung 11.6 Das Dialogfeld Datei entfernen



a) Zu entfernende Datei.

Eine Datei läßt sich wie folgt entfernen:

1. Wählen Sie den Befehl **Datei entfernen** aus dem Menü **Datei**.
2. Wählen Sie den Namen des/der zu entfernenden Moduls/e, Include-Dateien, Dokuments/e aus.
3. Drücken Sie die EINGABETASTE.

Das Modul wurde aus dem Arbeitsspeicher entfernt, ohne als Datei gelöscht zu werden. Mit Hilfe des Befehls **Datei entfernen** können sämtliche Dateitypen, sei es Modul, Include-Datei oder Dokument, aus dem Arbeitsspeicher gelöscht werden.

Siehe auch

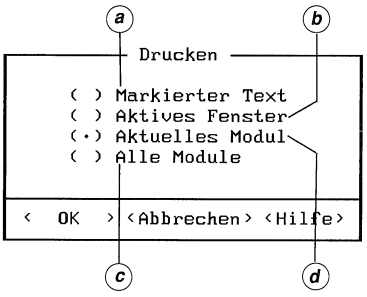
Abschnitt 11.1, "Der Befehl Neues Programm", und Abschnitt 16.7.2, "Die MAK-Datei".

11.10 Der Befehl Drucken

Mit Hilfe des Befehls **Drucken** aus dem Menü **Datei** können Sie ausgewählten Text, im aktiven Fenster vorhandenen Text, das gesamte aktuelle Modul oder alle aktuell geladenen Module des Programms ausdrucken. Die Anwendung des Befehls **Drucken** setzt einen am Anschluß LPT1 des Computers angeschlossenen Drucker voraus.

Bei Auswahl des Befehls **Drucken** aus dem Menü **Datei** erscheint das in Abbildung 11.7 dargestellte Dialogfeld. Wählen Sie die zum Ausdrucken der Datei geeignete Option aus dem Optionsfeld **Drucken** aus.

Abbildung 11.7 Das Dialogfeld Drucken



a) Ausdrucken von ausgewähltem Text.
b) Ausdrucken des im aktiven Fenster vorhandenen Texts.
c) Ausdrucken aller geladenen Module.
d) Ausdrucken des im aktiven Fenster vorhandenen Moduls.

11.11 Der Befehl Betriebssystem (nur im vollständigen Menü)

Der Befehl **Betriebssystem** erlaubt Ihnen, zeitweilig zu der DOS-Befehlsebene zurückzukehren, aus der andere Programme und DOS-Befehle ausgeführt werden können. Während dieses Vorgangs bleibt QuickBASIC im Arbeitsspeicher, so daß Sie an die ursprüngliche Stelle im Programm zurückkehren können, ohne es erneut zu laden.

Vor Ausführung des Befehls **Betriebssystem** muß QuickBASIC die Datei *command.com* finden, nach der es zuerst im Verzeichnis, das in der COMSPEC-Umgebungsvariablen angegeben ist, und dann im aktuellen Verzeichnis sucht. Weitere Informationen zu *command.com* und COMSPEC sind dem DOS-Handbuch zu entnehmen.

Kehren Sie wie folgt von der DOS-Betriebssystemebene zu QuickBASIC zurück:

1. Geben Sie `Exit` ein.
2. Drücken Sie die EINGABETASTE.

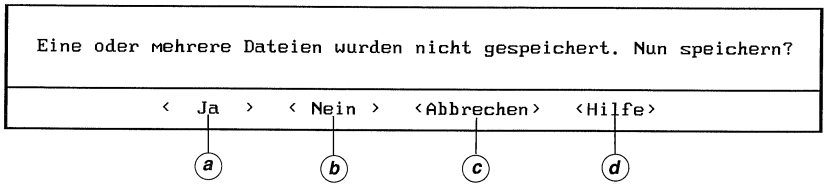
Die QuickBASIC-Anzeige erscheint unverändert auf dem Bildschirm.

11.12 Der Befehl Ende

Der Befehl **Ende** aus dem Menü **Datei** löscht QuickBASIC aus dem Arbeitsspeicher und kehrt zur DOS-Befehlsebene zurück.

Wenn Sie ein neues oder geändertes Programm verlassen, ohne es vorher zu speichern, erscheint das in Abbildung 11.8 gezeigte Dialogfeld.

Abbildung 11.8 Das Dialogfeld Ende



a) *Speichert die Datei(en).*
b) *Kehrt zu DOS zurück, ohne das Programm zu speichern.*
c) *Bleibt in der QuickBASIC-Umgebung.*
d) *Bietet Hilfe zum Dialogfeld.*

Nach Aufruf des Dialogfeldes **Ende** können Sie QuickBASIC auf folgende Arten beenden:

1. Wählen Sie **Ja** aus, oder drücken Sie die EINGABETASTE, um das Programm in seinem aktuellen Zustand zu speichern und zum Betriebssystem zurückzukehren.

   Wenn es sich um ein neues, noch unbenanntes Programm handelt, zeigt QuickBASIC ein Dialogfeld an, das dem Dialogfeld **Speichern** gleicht. Weitere Einzelheiten zu diesem Dialogfeld entnehmen Sie bitte Abschnitt 11.4.
2. Wählen Sie **Nein** aus, oder geben Sie N ein, um zu DOS zurückzukehren, ohne die Datei in ihrem aktuellen Zustand zu speichern.

Wenn Sie die QuickBASIC-Umgebung doch nicht verlassen möchten, drücken Sie ESC.

Siehe auch

Abschnitt 11.5, "Der Befehl Speichern unter", und Abschnitt 11.11, "Der Befehl Betriebssystem".

12 Verwendung des Editors

Dieses Kapitel befaßt sich mit der Eingabe und Bearbeitung von Programmtext mit Hilfe der Editierbefehle von QuickBASIC. Vor Beginn dieses Kapitels sollten Ihnen der QuickBASIC-Bildschirm und die in Kapitel 10, "Die Anwendung von QuickBASIC", beschriebenen Techniken geläufig sein.

Dieses Kapitel behandelt folgende Themen:

- Durchführen einfacher Editierfunktionen, wie Texteingabe und Bewegen des Cursors.
- Löschen und Einfügen von Textpassagen.
- Versetzen und Kopieren von Textblöcken.
- Suchen und Ersetzen von Zeichen, Wörtern oder Wortgruppen.
- Kopieren von Textpassagen aus anderen Dateien.

12.1 Die Texteingabe

Einem Text können neue Zeichen entweder durch Einfügen oder durch Überschreiben hinzugefügt werden. Im "Einfügemodus" fügt der QuickBASIC-Editor jedes neue Zeichen links von der Cursorstelle ein. Im "Überschreibemodus" ersetzt jedes neue Zeichen das an der Cursorstelle stehende Zeichen. Sie können durch Betätigen der EINFÜGETASTE bzw. der Tastenkombination STRG+V zwischen Einfüge- und Überschreibemodus umschalten. Im Einfügemodus hat der Cursor die Form eines blinkenden Unterstreichungszeichens, während er im Überschreibemodus die Form eines blinkenden Feldes annimmt.

Bei Eingabe eines QuickBASIC-Programms kann eine Zeile durch Betätigen der EINGABETASTE am Zeilenende oder durch Fortbewegen des Cursors von dieser Zeile (anhand einer Richtungstaste) abgeschlossen werden. Dies teilt QuickBASIC mit, daß eine oder mehrere BASIC-Anweisungen abgeschlossen sind. Wenn der Befehl **Syntaxüberprüfung** aus dem Menü **Optionen** (nur im vollständigen Menü verfügbar) aktiviert ist, überprüft QuickBASIC die Zeile auf die Richtigkeit der Anweisungssyntax.

12.2 Die Textauswahl

Vor Durchführen von Editierfunktionen, wie Kopieren, Löschen oder die Manipulieren Textblöcken, muß zunächst die zu bearbeitende Textpassage ausgewählt (hervorgehoben) werden. Dieser Vorgang zeigt sowohl dem Computer als auch dem Anwender die Textpassage, die durch den nächsten Befehl verändert wird.

Text läßt sich durch Betätigen der RICHTUNGSTASTEN bei niedergedrückter UMSCHALTTASTE oder einer der Tastenkombinationen zur Textbearbeitung (siehe Abschnitt 10.2.2, "Einsatz der Tastenkombinationen zur direkten Befehlseingabe") auswählen. Drücken Sie beispielsweise die Tastenkombination UMSCHALT+ENDE, um den Text von der Cursorstelle bis zum Ende der Zeile auszuwählen. Durch Betätigen einer RICHTUNGSTASTE wird der Text wieder normal dargestellt.

Wichtig Werden Zeichen eingegeben, während Text ausgewählt ist, wird dieser durch die eingegebenen Zeichen ersetzt.

12.3 Die Texteinrückung

Texteinrückung verbessert die Lesbarkeit und das strukturierte Erscheinungsbild des Programms und kann auch bei der Fehlerbeseitigung behilflich sein. Einzelne Textzeilen können mit Hilfe der LEERTASTE oder der TAB-TASTE eingerückt werden. Ein Textblock läßt sich durch Auswahl der gewünschten Zeilen und Betätigen der TAB-TASTE einrücken.

Die Einrückung gehört zu den wesentlichen Bestandteilen eines wohlstrukturierten Programms, das gut lesbar, logisch und selbsterklärend sein soll. Die von QuickBASIC gebotenen Einrückungskontrollen werden in Tabelle 12.1 aufgeführt.

Tabelle 12.1 Die Einrückungskontrollen von QuickBASIC

Tastenkombinationen	***Beschreibung***
(Automatisch)	Rückt jede neue Zeile entsprechend der aktuellen Einrückungsebene ein.
POS1	Bricht die automatische Einrückung auf einer Leerzeile ab und bringt den Cursor zum linken Bildschirmrand.
RÜCKTASTE	Setzt die Einrückung um eine Ebene zurück.
STRG+Q+S	Bringt den Cursor zum linken Ende einer Zeile, die eingerückten Text enthält.
UMSCHALT+TAB	Entfernt führende Leerstellen einer Einrückungsebene. Im Fall eines ausgewählten Textblocks wird der gesamte Block nach links verschoben.

Der Standard-Tabulatorabstand beträgt acht Leerstellen und kann wie folgt geändert werden:

1. Wählen Sie den Befehl **Bildschirm** aus dem Menü **Optionen** aus.
2. Wählen Sie die Bildschirmoption **Tab-Abstand** aus.
3. Geben Sie den neuen Tabulatorabstand ein.
4. Drücken Sie die EINGABETASTE.

QuickBASIC verwendet einzelne Leerstellen (anstelle des literalen Tabulatorzeichens ASCII 9) zur Darstellung der Einrückungsebenen. Anhand der Option **Tab-Abstand** des Dialogfeldes **Bildschirm** aus dem Menü **Optionen** wird die Leerstellenanzahl pro Einrückungsebene festgelegt.

Es gibt Texteditoren, die beim Speichern von Textdateien mehrere Leerstellen mit Hilfe der literalen Tabulatorzeichen darstellen. Wenn Sie mit Dateien arbeiten, die mit einem solchen Editor erstellt wurden, oder wenn Sie vermuten, daß Ihr Texteditor literale Tabulatorzeichen benutzt, empfiehlt es sich, Anhang B, "Unterschiede zu früheren BASIC-Versionen", aus *Programmieren in BASIC* durchzulesen.

Siehe auch

Abschnitt 20.1, "Der Befehl Bildschirm".

12.4 Verwendung von Punkten zur Textmarkierung

Während der Arbeit an mehreren Teilen eines umfangreichen Programms können Sie QuickBASIC anweisen, sich eine bestimmte Stelle im Programm durch Festlegen eines "Punktes zur Textmarkierung" zu merken. Dieses Verfahren erlaubt dem Programmierer, von einem Markierungspunkt zum anderen zu springen. Es können bis zu vier, von 0 bis 3 gekennzeichnete Punkte zur Textmarkierung an beliebiger Stelle im Programm festgesetzt werden.

Die Punkte zur Textmarkierung können wie folgt eingesetzt werden:

- Bringen Sie den Cursor zu der zu markierenden Zeile und drücken Sie STRG+K und *n*, um den Markierungspunkt *n* festzulegen.
- Drücken Sie STRG+Q und *n*, um sich zum Markierungspunkt *n* zu bewegen.

12.5 Der intelligente Editor

Die Version 4.5 von QuickBASIC vereint Texteditor, Compiler und Debugger in einem einzigen "intelligenten Editor". Der intelligente Editor weist besondere Eigenschaften auf, die die Eingabe und Bearbeitung von BASIC-Programmen erleichtern. Wenn der intelligente Editor eingeschaltet ist, führt QuickBASIC nachstehende Aktionen bei Eingabe jeder Zeile durch:

- Überprüfen der Zeile auf Syntaxfehler.
- Geeignetes Formatieren der Zeile.
- Übersetzen der Zeile in eine ausführbare Form, wenn die Syntax korrekt ist. Mit anderen Worten bedeutet das, daß die Zeile sofort nach Eingabe ausführbar ist.

12.5.1 Einschalten des intelligenten Editors

Der QuickBASIC-Editor wird meistens zur Eingabe und Bearbeitung von Programmtext benutzt. Wenn Sie QuickBASIC starten und mit der Eingabe beginnen, nimmt QuickBASIC an, daß Sie BASIC-Anweisungen schreiben, und schaltet den intelligenten Editor zu diesem Zweck ein. Der intelligente Editor wird auch bei der Durchführung folgender Vorgänge eingeschaltet:

- Auswahl der Befehle **Neues Programm** oder **Programm laden** aus dem Menü **Datei**.
- Auswahl der Befehle **Datei erstellen** oder **Lade Datei** aus dem Menü **Datei** und der Option **Modul** oder **Include** aus dem entsprechenden Dialogfeld.

Es kann gelegentlich vorkommen, daß Sie eine Textdatei bearbeiten müssen, die nicht aus BASIC-Anweisungen besteht, wie zum Beispiel die MAK-Datei von QuickBASIC oder ein Dokument. Zu diesem Zweck können Sie den intelligenten Editor ausschalten und die Umgebung wie ein normales Textverarbeitungssystem benutzen. Sie brauchen nur den Befehl **Datei erstellen** oder **Lade Datei** aus dem Menü **Datei** und dann die Option **Dokument** im Dialogfeld auszuwählen.

12.5.2 Die automatische Syntaxüberprüfung

Wenn der intelligente Editor eingeschaltet ist, überprüft er die Syntax jeder Zeile bei Eingabe. Ein Syntaxfehler weist darauf hin, daß eine Anweisung in der Zeile für QuickBASIC sinnlos ist. Nachstehende Zeile verursacht zum Beispiel einen Syntaxfehler, da das Schlüsselwort **GOTO** kein gültiges Argument für die **PRINT**-Anweisung darstellt.

```
PRINT GOTO
```

Beim Auftreten eines Fehlers zeigt QuickBASIC eine Fehlermeldung an. Löschen Sie die Fehlermeldung durch Betätigen der ESC- oder LEERTASTE, und positionieren Sie den Cursor auf der Stelle, an der der Fehler aufgetreten ist. Syntaxfehler müssen vor Ausführung des Programms verbessert werden.

Hinweis Beim Löschen einer Syntaxfehlermeldung wird der Cursor auf die fehlerverursachende Anweisung plaziert. Fordern Sie die On-Line-Hilfe für die Syntax der betreffenden Anweisung durch Drücken der Taste F1 an.

Bestimmte Tippfehler werden bei Eingabe nicht als Syntaxfehler erkannt. Bei Eingabe folgender Zeile anstelle des Schlüsselwortes **PRINT** erscheint beispielsweise keine Fehlermeldung:

```
pront
```

Obwohl es sich nicht um eine gültige BASIC-Anweisung handelt, erkennt QuickBASIC den Fehler erst beim Versuch, das Programm auszuführen. Bis zu diesem Zeitpunkt wird die Anweisung als ein Aufruf einer **SUB**-Prozedur mit dem Namen `pront` interpretiert.

Die Syntaxüberprüfung wird fortlaufend durchgeführt, es sei denn, Sie wählen die Befehle **Datei erstellen** oder **Lade Datei** mit der Option **Dokument** aus. Mit Hilfe des Befehls **Syntaxüberprüfung** aus dem Menü **Optionen** wird die Syntaxüberprüfung ein- und ausgeschaltet. Ein neben diesem Befehl stehender Punkt deutet darauf hin, daß die Syntaxüberprüfung eingeschaltet ist.

12.5.3 Die Fehlermeldungen

QuickBASIC übersetzt fehlerfreie Anweisungen bei Eingabe in ausführbaren Code. Beim Auftreten von Syntaxfehlern zeigt QuickBASIC ein Dialogfeld mit der entsprechenden Fehlermeldung an. Wenn QuickBASIC eine Ihnen unverständliche Fehlermeldung ausgibt, können Sie weitere Informationen zu der jeweiligen Fehlermeldung durch Auswählen der Hilfsbefehlsfläche im Dialogfeld anfordern. Um nähere Informationen über die Anwendung der fehlerverursachenden Anweisung zu erhalten, wählen Sie **OK** im Dialogfeld aus, und führen Sie einen der folgenden Arbeitsschritte durch:

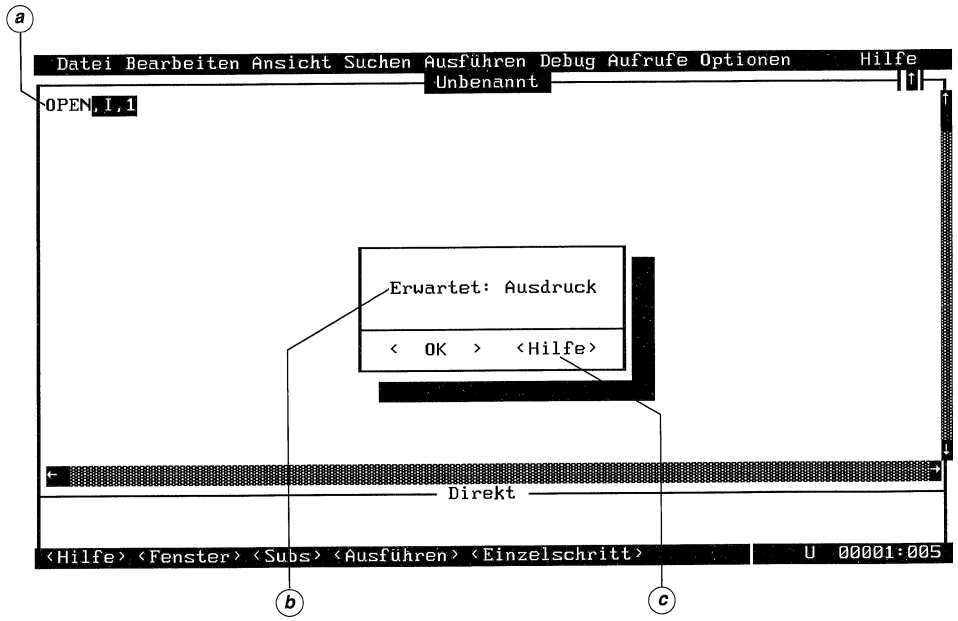
- Wählen Sie den Befehl **Begriff** aus dem Menü **Hilfe**.
- Drücken Sie F1.

Sehr oft erscheint die allgemeine Fehlermeldung `Syntaxfehler`, einige Fehlermeldungen enthalten jedoch genauere Angaben. Bei Eingabe folgender Zeile, in der ein Dateiname fehlt:

```
OPEN,I,1,
```

gibt QuickBASIC zum Beispiel, wie in Abbildung 12.1 gezeigt, die Fehlermeldung `Erwartet: Ausdruck` aus:

Abbildung 12.1 Syntaxfehlermeldung



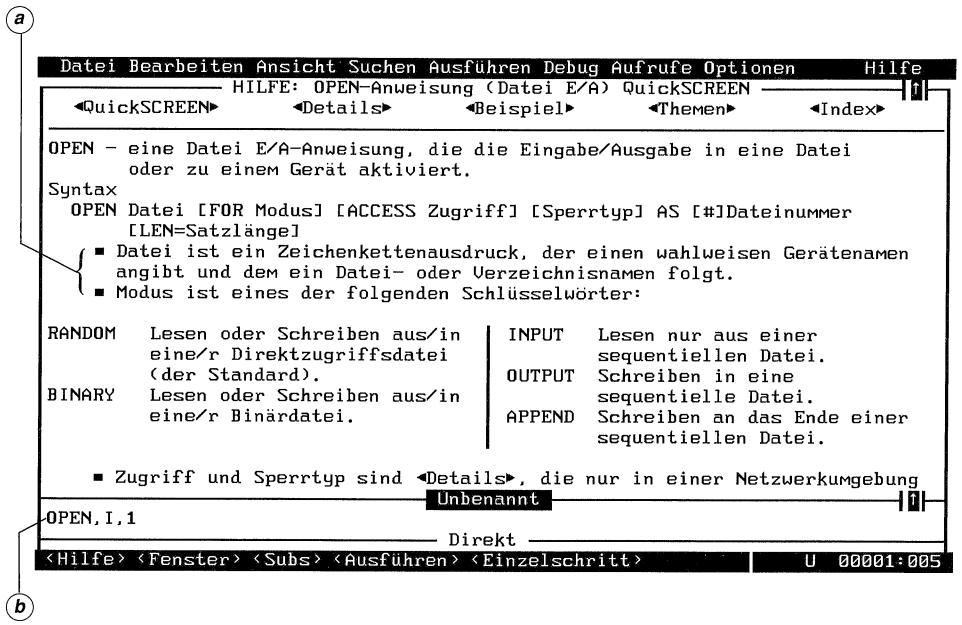
a) *Unvollständige Anweisung.*
b) *Typ des Syntaxfehlers.*
c) *Hilfe für den Fehler.*

Im oben erwähnten Fall fordern Sie die On-Line-Hilfe für die Anweisung `OPEN` wie folgt an:

1. Wählen Sie **OK** im Dialogfeld aus.
2. Positionieren Sie den Cursor auf dem Schlüsselwort `OPEN`.
3. Drücken Sie die Taste F1.

Die Anzeige der On-Line-Hilfe erscheint wie in Abbildung 12.2 dargestellt auf dem Bildschirm:

Abbildung 12.2 Hilfe für das Schlüsselwort OPEN



a) Verschiedene Anwendungen des Schlüsselwortes OPEN.

b) Programmtext.

In der in Abbildung 12.2 dargestellten Anzeige können Sie die gewünschte Anwendung durch Positionieren des Cursors in der entsprechenden Zeile und Betätigen der Taste F1 auswählen. Wie Sie feststellen, bleibt der Programmtext während der Verwendung der On-Line-Hilfe auf dem Bildschirm bestehen. Verlassen Sie die On-Line-Hilfe durch Betätigen der ESC-TASTE.

Siehe auch

Kapitel 8, "Zusammenfassung der Anweisungen und Funktionen", und Anhang I, "Fehlermeldungen", in *Programmieren in BASIC*.

12.5.4 Die automatische Formatierung

Außer der Überprüfung von Syntaxfehlern standardisiert der intelligente Editor auch das Format jeder Zeile bei der Eingabe, indem er folgende Änderungen vornimmt:

- Umwandlung aller BASIC-Schlüsselwörter in Großbuchstaben.
- Einfügen von Leerstellen vor und nach allen BASIC-Operatoren.
- Beibehaltung einer konsequenten Schreibweise in allen Verweisen auf Variablen oder Prozeduren.
- Einfügen von Satzzeichen falls erforderlich. Der intelligente Editor setzt beispielsweise Semikolons zwischen aufeinanderfolgenden Elementen einer **PRINT**-Anweisung selbständig ein und ergänzt, falls dieses vom Programmierer versäumt wurde, das Argument einer **PRINT**-Anweisung automatisch mit einem Anführungszeichen.

QuickBASIC erfaßt die Namen sämtlicher Variablen und Prozeduren des Programms und ändert die Groß- und Kleinschreibung aller Vorkommen einer Variablen- oder Prozedurbezeichnung gemäß der zuletzt eingegebenen Version. Wenn Sie zum Beispiel einer Variablen zuerst den Namen `MeinVar` (mit den Großbuchstaben `M` und `V`) geben und diese Bezeichnung später als `meinvar` (nur Kleinbuchstaben) eingeben, ändert QuickBASIC alle Vorkommen von `MeinVar` in `meinvar` um.

12.6 Eingabe von Sonderzeichen

QuickBASIC verfügt über Prozeduren für die Eingabe literaler Darstellungen von Sonderzeichen, einschließlich werthöherer ASCII-Zeichen und der mit Steuerzeichenfolgen verbundenen literalen ASCII-Zeichen. (Im Allgemeinen erhöht die Eingabe solcher Literalzeichen die Lesbarkeit des Programms, vermindert jedoch seine Datenaustauschbarkeit, die mit der **CHR$**-Funktion gesteuert wird. Siehe auch Abschnitt 9.5, "Zusammenfassung der Anweisungen und Funktionen zur Zeichenkettenverarbeitung", in *Programmieren in BASIC*.)

Werthöhere ASCII-Zeichen liegen im Dezimalbereich 128-255 und können direkt in einem QuickBASIC-Programm durch Eingabe des entsprechenden Dezimalcodes auf dem numerischen Tastenblock bei niedergedrückter ALT-TASTE eingegegeben werden. Die Eingabe des Dezimalcodes 205 bei niedergedrückter ALT-TASTE erzeugt zum Beispiel das Zeichen Doppelstrich.

Zahlreiche ASCII-Zeichen des unteren Wertebereichs zwischen 1 und 27 sind sowohl Literalzeichen, als auch mit Sonderfunktionen verbunden. Das durch die Tastenkombination ALT+25 erzeugte Zeichen stellt beispielsweise sowohl den Pfeil nach unten (↓), als auch die Steuersequenz ^Y (Löschen der aktuellen Zeile) dar. Wenn Sie das Zeichen für den Pfeil nach unten einer **PRINT**-Anweisung übergeben möchten, können Sie den Dezimalcode 25 als Argument der **CHR$**-Funktion angeben. Näheres über die dezimalen Werte der ASCII-Zeichen ist Anhang D, "Tastaturabfragecodes und ASCII-Zeichencodes", aus *Programmieren in BASIC* zu entnehmen. Es ist jedoch auch möglich, das Literalzeichen selbst durch folgende Methode einzugeben:

1. Betätigen Sie die Taste P bei niedergedrückter STRG-TASTE.

   Anschließend erscheint das Zeichen `^P` in der QuickBASIC-Statuszeile. Das gewünschte Literalzeichen kann jetzt durch Betätigen einer STRG-Tastenkombination erzeugt werden.

2. Betätigen Sie die dem gewünschten Literalzeichen entsprechende Taste bei niedergedrückter STRG-TASTE.

   Dadurch wird nicht die Steuerfunktion ausgeführt, sondern es erscheint an der Cursorstelle das literale ASCII-Zeichen.

Geben Sie zum Beispiel das Zeichen Pfeil nach unten (↓) innerhalb der Anführungsstriche einer **PRINT**-Anweisung durch Drücken der Tastenkombination STRG+P+Y ein. Beim Versuch, dieses Zeichen durch die Tastenkombination STRG+Y oder ALT+25 einzugeben, wird die aktuelle Zeile gelöscht, wenn nicht zuvor `^P` eingegeben wurde. Diese Technik darf jedoch nicht bei Eingabe folgender Zeichen angewandt werden: ^J, ^L, ^M, ^U. Eine komplette Liste der ASCII-Zeichen ist im Anhang D, "Tastaturabfragecodes und ASCII-Zeichencodes", des Handbuchs *Programmieren in BASIC* zu finden.

12.7 Zusammenfassung der Editierbefehle

Der als flexibles Programmierwerkzeug entwickelte Editor von QuickBASIC unterstützt eine Tastaturschnittstelle, die den Benutzern anderer Microsoft-Produkte wie Microsoft Word bereits bekannt ist. Den Anwendern von WordStar und ähnlichen Editoren sind zahlreiche Kombinationen der STRG-TASTE mit anderen Tasten geläufig. Tabelle 12.2 faßt sämtliche Editierbefehle von QuickBASIC zusammen.

Tabelle 12.2 Editierbefehle

Cursorbewegung	***Tasten des Tastenblocks***	***Tasten-kombinationen im WordStar-Stil***
Ein Zeichen nach links	NACH LINKS TASTE	STRG+S
Ein Zeichen nach rechts	NACH RECHTS TASTE	STRG+D
Ein Wort nach links	STRG+NACH LINKS	STRG+A
Ein Wort nach rechts	STRG+NACH RECHTS	STRG+F
Eine Zeile nach oben	NACH OBEN TASTE	STRG+E
Eine Zeile nach unten	NACH UNTEN TASTE	SRTG+X
Erste Einrückungsebene	POS1	--
Erste Spalte der aktuellen Zeile	--	STRG+Q+S
Anfang der nächsten Zeile	STRG+EINGABE	STRG+J
Ende der Zeile	ENDE	STRG+Q+D
Oberes Ende des Fensters	--	STRG+Q+E
Ende der Datei	--	STRG+Q+X
Anfang des/der Moduls/Prozedur	STRG+POS1	STRG+Q+R
Ende des/der Moduls/Prozedur	STRG+ENDE	STRG+Q+C
Setzen eines Markierungspunktes	--	STRG+K *n* *
Springen zum Markierungspunkt	--	STRG+Q *n* *

Einfügen	***Tasten des Tastenblocks***	***Tasten-kombinationen im WordStar-Stil***
Einfügemodus ein/aus	EINFG-TASTE	STRG+V
In der Zeile unterhalb	ENDE+EINGABE	--
In der Zeile oberhalb	--	POS1 STRG+N
Inhalt des Zwischenspeichers	UMSCHALT+EINFG	--
Tabulator an der Cursorstelle oder Anfang jeder ausgewählten Zeile	TAB-TASTE	STRG+I
Steuerzeichen an der Cursorstelle	--	STRG+P+ STRG+*Taste* **

Löschen	***Tasten des Tastenblocks***	***Tasten-kombinationen im WordStar-Stil***
Aktuelle Zeile (und Abspeichern im Zwischenspeicher)	--	STRG+Y
Bis Ende der Zeile (und Abspeichern im Zwischenspeicher)	--	STRG+Q+Y
Linkes Zeichen	RÜCKTASTE	STRG+H
Zeichen an der Cursorstelle	ENTF-TASTE	STRG+G
Wort	--	STRG+T
Ausgewählter Text (und Abspeichern im Zwischenspeicher)	UMSCHALT+ENTF	--
Ausgewählter Text (ohne Abspeichern im Zwischenspeicher)	ENTF-TASTE	STRG+G
Führende Leerstellen für eine Einrückungsebene der ausgewählten Zeilen	UMSCHALT+TAB	--

Kopieren	***Tasten des Tastenblocks***	***Tasten-kombinationen im WordStar-Stil***
Ausgewählten Text in den Zwischenspeicher	STRG+EINFG	--

Rollen	***Tasten des Tastenblocks***	***Tasten-kombinationen im WordStar-Stil***
Eine Zeile nach oben	STRG+NACH OBEN	STRG+W
Eine Zeile nach unten	STRG+NACH UNTEN	STRG+Z
Eine Seite nach oben	BILD NACH OBEN TASTE	STRG+R
Eine Seite nach unten	BILD NACH UNTEN TASTE	STRG+C
Ein Fenster nach links	STRG+BILD NACH OBEN	--
Ein Fenster nach rechts	STRG+BILD NACH UNTEN	--

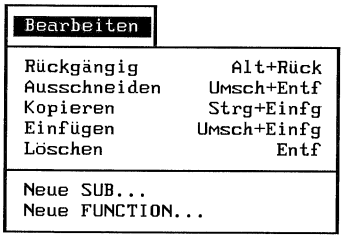
Auswählen	*Tasten des Tastenblocks*	*Tasten-kombinationen im WordStar-Stil*
Linkes Zeichen	UMSCHALT+NACH LINKS	--
Rechtes Zeichen	UMSCHALT+NACH RECHTS	--
Aktuelle Zeile ***	UMSCHALT+NACH UNTEN	--
Obere Zeile	UMSCHALT+NACH OBEN	--
Linkes Wort	UMSCHALT+STRG+ NACH LINKS	--
Rechtes Wort	UMSCHALT+STRG+ NACH RECHTS	--
Eine Bildschirmseite nach oben	UMSCHALT+BILD NACH OBEN	--
Eine Bildschirmseite nach unten	UMSCHALT+BILD NACH UNTEN	--
Eine Bildschirmseite nach links	UMSCHALT+STRG+ BILD NACH OBEN	--
Eine Bildschirmseite nach rechts	UMSCHALT+STRG+ BILD NACH UNTEN	--
Bis Anfang des/der Moduls/Prozedur	UMSCHALT+STRG+ POS1	--
Bis Ende des/der Moduls/Prozedur	UMSCHALT+STRG+ ENDE	--

* für *n* ist zum Kennzeichnen des gewünschten Markierungspunktes eine Zahl zwischen 0 und 3 einzusetzen. (Vor Eingabe von *n* die Tastenkombination STRG+K loslassen.)

** Nach der Tastenkombination STRG+P wird die STRG-TASTE und die gewünschte Taste betätigt.

*** Wählt die aktuelle Zeile nur dann aus, wenn sich der Cursor auf dem ersten Zeichen befindet. Andernfalls wird sowohl die aktuelle als auch die nachfolgende Zeile ausgewählt.

13 Das Menü Bearbeiten



Das Menü **Bearbeiten** steuert die zum Erstellen und Manipulieren des Programmcodes (Texts) verwendeten Funktionen. Mit Hilfe dieses Menüs lassen sich Textpassagen ausschneiden, kopieren und einfügen, die zuletzt vorgenommene Änderung rückgängig machen und **SUB**- oder **FUNCTION**-Prozeduren erstellen.

Das Menü **Bearbeiten** umfaßt folgende Befehle:

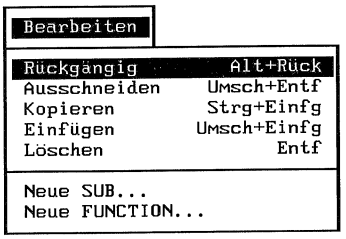
- **Rückgängig**. Macht den letzten Befehl oder die letzte Aktion rückgängig und stellt den vorhergehenden Zustand der Zeile wieder her (nur im vollständigen Menü).
- **Ausschneiden**. Löscht den ausgewählten Text und kopiert diesen in den Zwischenspeicher.
- **Kopieren**. Bringt eine Kopie des ausgewählten Texts in den Zwischenspeicher.
- **Einfügen**. Fügt den Inhalt des Zwischenspeichers an der Cursorstelle ein.
- **Löschen**. Löscht den ausgewählten Text endgültig (nur im vollständigen Menü).
- **Neue SUB**. Erstellt eine neue **SUB**-Prozedur in einem eigenen Fenster (nur im vollständigen Menü).
- **Neue FUNCTION**. Erstellt eine neue **FUNCTION**-Prozedur in einem eigenen Fenster (nur im vollständigen Menü).

13.1 Der Zwischenspeicher

Der Editor von QuickBASIC benutzt einen Zwischenspeicher, der Teil des Arbeitsspeichers ist und zum zeitweiligen Speichern von ausgeschnittenen oder kopierten Textblöcken bestimmt ist. Das Prinzip des Zwischenspeichers ist Ihnen wahrscheinlich bekannt, wenn Sie bereits ein Textverarbeitungssystem zum Ausschneiden, Kopieren und Einfügen von Textpassagen verwendet haben.

Beim Verschieben, Löschen oder Duplizieren von Textabschnitten mit Hilfe des Befehls **Ausschneiden** oder **Kopieren** wird der betreffende Text im Zwischenspeicher gespeichert. Der Inhalt des Zwischenspeichers kann danach anhand des Befehls **Einfügen** beliebig oft an anderen Stellen eingefügt werden, ohne den im Zwischenspeicher vorhandenen Text zu verändern. Bei erneuter Anwendung der Befehle **Ausschneiden** oder **Kopieren** wird jedoch der Inhalt des Zwischenspeichers durch den neuen Text ersetzt.

13.2 Der Befehl Rückgängig (nur im vollständigen Menü)



Der Befehl **Rückgängig** aus dem Menü **Bearbeiten** macht alle in der aktuellen Zeile ausgeführten Änderungen rückgängig. Wenn jedoch der Cursor in der Zwischenzeit von der Zeile entfernt wurde, kann dieser Befehl den früheren Zustand der Zeile nicht mehr herstellen.

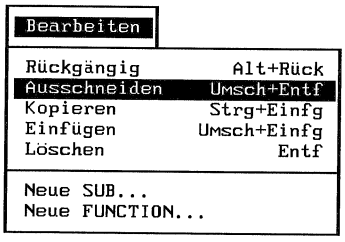
Der Befehl **Rückgängig** wirkt sich auf eine Zeile nur dann aus, wenn Änderungen vorgenommen wurden. Andernfalls ist dieser Befehl wirkungslos.

Hinweis Der Befehl **Rückgängig** kann einen mit der Tastenkombination STRG+Y gelöschten Text nicht mehr wiederherstellen, da diese Kombination eine ganze Zeile vom Bildschirm entfernt. In diesem Fall enthält jedoch der Zwischenspeicher eine Kopie der Zeile, sodaß sie mit Hilfe des Befehls **Einfügen** wiederhergestellt werden kann.

Tastenkombination zur direkten Befehlseingabe

ALT+RÜCK

13.3 Der Befehl Ausschneiden



Der Befehl **Ausschneiden** entfernt den ausgewählten Text vom Bildschirm und kopiert diesen in den Zwischenspeicher. Der Befehl **Ausschneiden** ist nur verfügbar, wenn zuvor ein Textblock markiert wurde.

Zeilen und Textblöcke lassen sich durch die kombinierte Anwendung der Befehle **Ausschneiden** und **Einfügen** wie folgt verschieben:

1. Markieren Sie die Textpassage. Sie kann aus einer beliebigen Anzahl Zeilen bestehen.
2. Führen Sie den Befehl **Ausschneiden** aus dem Menü **Bearbeiten** aus.
3. Bringen Sie den Cursor an die Stelle, an der Sie den Textblock einfügen wollen.
4. Führen Sie den Befehl **Einfügen** aus dem Menü **Bearbeiten** aus.

Dieses Verfahren kann zum Verschieben ganzer Programmblöcke von einem Abschnitt zu einem anderen angewandt werden und erweist sich dadurch als sehr zeitsparend beim Schreiben neuer Programme.

Hinweis Der Zwischenspeicher beinhaltet nur den zuletzt ausgeschnittenen oder kopierten Textabschnitt.

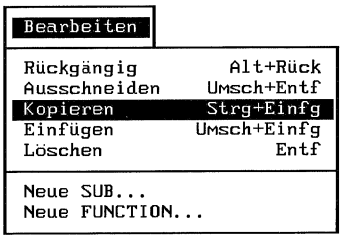
Tastenkombination zur direkten Befehlseingabe

UMSCHALT+ENTF

Siehe auch

Abschnitt 13.4, "Der Befehl Kopieren", Abschnitt 13.5, "Der Befehl Einfügen", und Abschnitt 13.6, "Der Befehl Löschen".

13.4 Der Befehl Kopieren



Dieser Befehl kopiert den ausgewählten Text in den Zwischenspeicher, wobei der Originaltext unverändert auf dem Bildschirm beibehalten wird. Durch diesen Vorgang wird der vorhergehende Inhalt des Zwischenspeichers durch den neuen Text ersetzt.

Die kombinierte Anwendung der Befehle **Kopieren** und **Einfügen** dupliziert Abschnitte oder Zeilen des Programmcodes. Mit Hilfe nachstehender Arbeitsschritte kann beispielsweise die Erstellung ähnlicher **SUB**-Prozeduren beschleunigt werden:

1. Markieren Sie den zu duplizierenden Text.
2. Führen Sie den Befehl **Kopieren** aus dem Menü **Bearbeiten** aus.
3. Bringen Sie den Cursor an die Stelle, an der der Text eingefügt werden soll.
4. Führen Sie den Befehl **Einfügen** aus dem Menü **Bearbeiten** aus.
5. Passen Sie den kopierten Text der neuen Zeile, Prozedur oder dem neuen Programmabschnitt an.

Hinweis Der Zwischenspeicher enthält nur den zuletzt ausgeschnittenen oder kopierten Text.

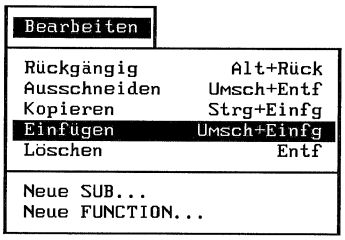
Tastenkombination zur direkten Befehlseingabe

STRG+EINFG

Siehe auch

Abschnitt 13.3, "Der Befehl Ausschneiden", und Abschnitt 13.5, "Der Befehl Einfügen".

13.5 Der Befehl Einfügen



Der Befehl **Einfügen** fügt eine Kopie des im Zwischenspeicher vorhandenen Texts rechts von der Cursorstelle ein.

Der Befehl ist nur verfügbar, wenn der Zwischenspeicher einen Textabschnitt enthält, und ersetzt markierten Text durch den Inhalt des Zwischenspeichers. Ist kein Text markiert, fügt dieser Befehl den Inhalt des Zwischenspeichers rechts vom Cursor oder oberhalb des Cursors ein, falls der Inhalt des Cursors eine Zeile überschreitet.

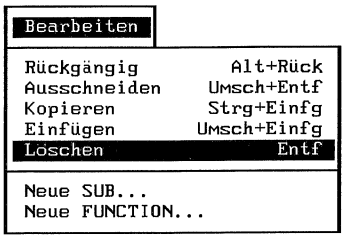
Tastenkombination zur direkten Befehlseingabe

UMSCHALT+EINFG

Siehe auch

Abschnitt 13.3, "Der Befehl Ausschneiden", und Abschnitt 13.4, "Der Befehl Kopieren".

13.6 Der Befehl Löschen (nur im vollständigen Menü)



Der Befehl **Löschen** entfernt den markierten Text endgültig aus der Datei.

Der gelöschte Text wird nicht in den Zwischenspeicher kopiert und kann nicht mit Hilfe des Befehls **Einfügen** ersetzt werden. Ist kein Text ausgewählt, wird durch Betätigen der ENTF-TASTE das Zeichen rechts vom Cursor gelöscht.

Warnung Mit dem Befehl **Löschen** ist Vorsicht geboten, da gelöschter Text anhand des Befehls **Rückgängig** nur unmittelbar nach dem Löschvorgang wiederhergestellt werden kann.

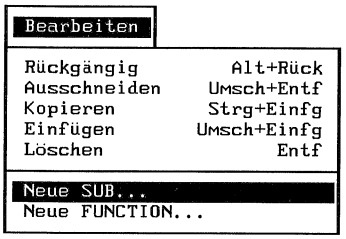
Taste zur direkten Befehlseingabe

ENTF

Siehe auch

Abschnitt 13.2, "Der Befehl Rückgängig", und Abschnitt 13.3, "Der Befehl Ausschneiden".

13.7 Der Befehl Neue SUB (nur im vollständigen Menü)



Der Befehl **Neue SUB** aus dem Menü **Bearbeiten** erstellt und benennt eine neue **SUB**-Prozedur als Bestandteil des im aktiven Fenster vorhandenen Moduls. Beim Speichern des Programms werden sämtliche neue Prozeduren dem Modul hinzugefügt.

SUB-Prozeduren können im Hauptmodul oder in anderen Modulen definiert und von einem Modul in das andere mit Hilfe des Befehls **SUBs** aus dem Menü **Ansicht** verschoben werden.

SUB-Prozeduren eignen sich für wiederholte Operationen, einschließlich jener, die übergebene Informationen (Argumente) verändern. Sie können beispielsweise die **SUB**-Prozedur `ZentrText` zum Zentrieren einer Textpassage auf dem Bildschirm definieren. Bei Eingabe nachstehender Zeile im Programm:

```
CALL ZentrText ("Stundenlohn")
```

erscheint der Ausdruck `Stundenlohn` in der Mitte des Bildschirms (der **SUB**-Code selbst wurde hier ausgelassen).

Wichtig Der Name einer Prozedur kann nur ein einziges Mal in einem Programm definiert werden. Andernfalls tritt ein Fehler auf.

Siehe auch

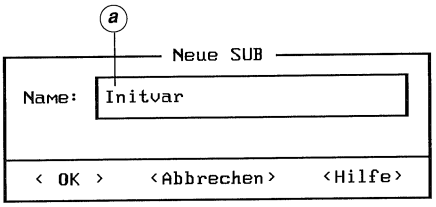
Kapitel 5, "Das Programm QKARTEN", aus diesem Handbuch und Kapitel 2, "Prozeduren: Unterprogramme und Funktionen", aus *Programmieren in BASIC*.

13.7.1 Erstellen einer neuen SUB-Prozedur

Eine neue **SUB**-Prozedur läßt sich wie folgt erstellen:

1. Wählen Sie **Neue SUB** aus dem Menü **Bearbeiten** aus. Es erscheint das Dialogfeld **Neue SUB**.
2. Geben Sie den Namen der neuen **SUB**-Prozedur, wie in Abbildung 13.1 gezeigt, im Textfeld ein.

Abbildung 13.1 Das Dialogfeld Neue SUB



a) Neuer SUB-Name.

3. Drücken Sie die EINGABETASTE.

Der Befehl **Neue SUB** öffnet ein neues Fenster und stellt die Anweisungen **SUB** und **END SUB** bereit. Nach der Codeeingabe können Sie mit Hilfe des Befehls **SUBs** aus dem Menü **Ansicht** zum Modul-Ebenen-Code, zu einem anderen Modul oder zu einer anderen Prozedur zurückkehren.

Bei Eingabe von `SUB` *Subname* beginnt QuickBASIC die Prozedur sofort, ohne zuerst das Menü oder das Dialogfeld zu öffnen.

Beim Beginn einer neuen "Prozedur" (der Sammelbegriff für **SUB**- und **FUNCTION**-Prozeduren) durch Eingabe von **SUB** oder **FUNCTION** und des Prozedurnamens oder anhand des Befehls **Neue SUB** oder **Neue FUNCTION** aus dem Menü **Bearbeiten** zeigt QuickBASIC die Prozedur im Arbeitsfenster an.

13.7.2 Standard-Datentypen für Prozeduren

Der von der Anweisung **DEF***typ* bestimmte Datentyp bezieht sich auf den einem bestimmten Namenbereich zugeordneten Datentyp. Der allgemeine Begriff **DEF***typ*-Anweisung umfaßt die datendefinierenden Anweisungen **DEFINT** (Definition der Ganzzahlen), **DEFSNG** (Definition der Variablen einfacher Genauigkeit), **DEFDBL** (Definition der Variablen doppelter Genauigkeit), **DEFLNG** (Definition der langen Ganzzahlen) und **DEFSTR** (Definition der Zeichenketten). Nach der **DEF***typ*-Anweisung steht eine Folge von Buchstaben, die den von dieser Anweisung beeinflußten Namenbereich angeben. `DEFDBL M-Q` definiert zum Beispiel alle Variablen, deren Namen mit Buchstaben aus dem Bereich M bis Q beginnen, als Variablen doppelter Genauigkeit.

In einer neuen Prozedur fügt QuickBASIC eine **DEF***typ*-Anweisung automatisch am oberen Ende des Arbeitsfensters, vor der Zeile mit dem Schlüsselwort **SUB**, ein:

```
DEFtyp Buchstabe1-Buchstabe2
SUB SubName
END SUB
```

oder

```
DEFtyp Buchstabe1-Buchstabe2
FUNCTION FunctionName
END FUNCTION
```

Im vorhergehenden Beispiel geben `Buchstabe1` und `Buchstabe2` den Buchstabenbereich an, auf den sich die **DEF***typ*-Anweisung auswirkt, wobei `Buchstabe2` ausgelassen werden kann, falls nur ein einziger Buchstabe als spezifischer Datentyp definiert werden soll.

Die der Prozedur automatisch hinzugefügte **DEF***typ*-Anweisung hängt vom Bestehen solcher Anweisungen auf Modulebene ab. Falls keine **DEF***typ*-Anweisungen im Hauptmodul vorhanden sind, erstellt QuickBASIC die Prozedur ohne eine **DEF***typ*-Anweisung, wobei alle Variablen der neuen Prozedur standardmäßig als Zahlen einfacher Genauigkeit definiert werden. Bestehen dagegen auf Modulebene eine oder mehrere **DEF***typ*-Anweisungen, schließt QuickBASIC alle diese Anweisungen, mit Ausnahme von **DEFSNG**, in der neuen Prozedur ein. Alle Variablen, die außerhalb des von den **DEF***typ*-Anweisungen angegebenen Buchstabenbereichs liegen, werden standardmäßig als Variablen einfacher Genauigkeit definiert.

13.7.3 Ändern des Standardtyps einer Prozedur

Falls QuickBASIC eine unerwünschte **DEF***typ*-Anweisung einfügt, kann diese entfernt (wobei die Prozedur standardmäßig zur Prozedur einfacher Genauigkeit wird) oder durch Überschreiben geändert werden.

Sind im Hauptmodul keine **DEF***typ*-Anweisungen vorhanden, läßt QuickBASIC solche Anweisungen in neuen Prozeduren aus, wobei deren Variablen den Standardtyp einfache Genauigkeit erhalten. Sie können einer Prozedur neue **DEF***typ*-Anweisungen hinzufügen, indem Sie diese vor einer **SUB**- oder **FUNCTION**-Anweisung (im Einfügemodus) eingeben.

13.7.4 Speichern und Benennen von Prozeduren

Beim Speichern eines Programms erstellt QuickBASIC eine Vereinbarung für jede Prozedur am Anfang des Moduls, das die jeweilige Prozedur enthält. Eine Vereinbarung besteht aus dem Schlüsselwort **DECLARE**, dem das **SUB**- oder **FUNCTION**-Schlüsselwort, der Name der Prozedur und eine Liste der formalen Parameter folgt.

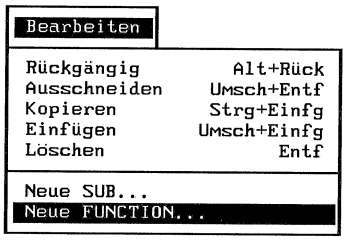
Zum Unterschied von Modulen werden Prozeduren nicht als selbständige Dateien gespeichert. Aus diesem Grund werden die Prozedurnamen nicht von DOS, sondern von den QuickBASIC-Namenskonventionen bestimmt. Prozeduren werden innerhalb des Moduls, in dem sie erstellt oder eingefügt wurden, gespeichert.

Die Regeln für die Benennung von Variablen gelten auch für Prozeduren. Der Name einer Prozedur muß ein gültiger, eindeutiger BASIC-Name sein und kann aus höchstens 40 Zeichen bestehen. Wird einer Prozedur der gleiche Name wie einer Variablen zugewiesen, erscheint die Fehlermeldung `Doppelte Definition`.

Siehe auch

Abschnitt 13.8, "Der Befehl Neue FUNCTION", und Abschnitt 14.1, "Der Befehl SUBs".

13.8 Der Befehl Neue FUNCTION (nur im vollständigen Menü)



Der Befehl **Neue FUNCTION** definiert (erstellt und benennt) eine neue **FUNCTION**-Prozedur als Bestandteil des im Arbeitsfenster vorhandenen Moduls oder Programms. Beim Speichern eines Moduls werden sämtliche neue Prozeduren im entsprechenden Modul eingegliedert.

FUNCTION-Prozeduren eignen sich für die wiederholte Berechnung ähnlicher Werte. Sie können beispielsweise eine eigene **FUNCTION**-Prozedur `Volumen (X)` definieren, die das Volumen einer Kugel mit dem Radius `X` berechnet. Das Programm ist dann in der Lage, das Volumen einer Kugel mit beliebigem Radius zu berechnen:

```
KugelVol = Volumen (Rad)    ' berechne Volumen einer Kugel
                            ' mit dem Radius Rad
```

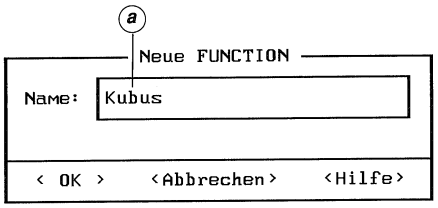
Bei der Berechnung des Volumens einer Kugel setzt QuickBASIC den Wert von `X` gleich `Rad`. Ist demnach die Berechnung des Volumens einer Kugel öfters erforderlich, kann sich diese **FUNCTION**-Prozedur als platz- und zeitsparend erweisen und das Debuggen vereinfachen.

Wichtig Der Name einer Prozedur darf nur ein einziges Mal im Programm definiert werden. Andernfalls tritt ein Fehler auf.

Eine neue **FUNCTION**-Prozedur läßt sich durch folgende Arbeitsschritte definieren:

1. Wählen Sie den Befehl **Neue FUNCTION** aus dem Menü **Bearbeiten**. Das Dialogfeld **neue FUNCTION** erscheint.
2. Geben Sie den Namen der neuen **FUNCTION**-Prozedur im Textfeld ein (siehe Abbildung 13.2).

Abbildung 13.2 Das Dialogfeld Neue FUNCTION



a) Name der neuen FUNCTION-Prozedur.

3. Drücken Sie die EINGABETASTE.

Anschließend öffnet QuickBASIC ein neues Fenster, in dem die Anweisungen **FUNCTION** und **END FUNCTION** bereits vorhanden sind. Kehren Sie nach Abschluß der **FUNCTION**-Prozedur anhand des Befehls **SUBs** aus dem Menü **Ansicht** zum aktuellen Modul zurück.

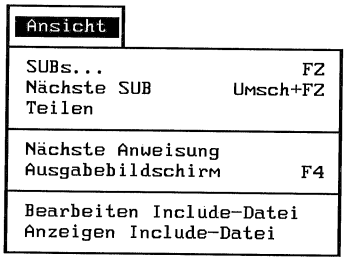
Direkte Eingabe

Geben Sie `FUNCTION` und den Prozedurnamen ein, um die oben aufgeführten Arbeitsschritte 1 bis 3 automatisch durchzuführen.

Siehe auch

Kapitel 5, "Das Programm QKARTEN", Abschnitt 13.7.2, "Standard-Datentypen für Prozeduren", Abschnitt 13.7.3, "Ändern des Standardtyps einer Prozedur", Abschnitt 13.7.4, "Speichern und Benennen von Prozeduren", Abschnitt 14.1, "Der Befehl SUBs", sowie Kapitel 2, "Prozeduren: Unterprogramme und Funktionen", aus *Programmieren in BASIC*.

14 Das Menü Ansicht

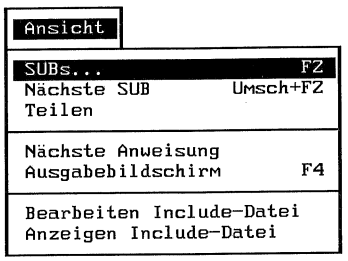


Das Menü **Ansicht** enthält die zum Bearbeiten von Programmteilen verfügbaren Optionen. Sie können damit auch **SUB**- und **FUNCTION**-Prozeduren, Module, Include-Dateien und Dokumente, die bearbeitet werden sollen, in das Arbeitsfenster bringen.

Das Menü **Ansicht** umfaßt folgende Befehle:

- **SUBs**. Bringt Prozeduren, Module, Include-Dateien und Dokumente in das Arbeitsfenster und verschiebt Prozeduren von einem Modul in das andere.
- **Nächste SUB**. Bringt die nächste Prozedur in alphabetischer Reihenfolge der Namen ins aktive Fenster (nur im vollständigen Menü).
- **Teilen**. Teilt das Arbeitsfenster horizontal auf (nur im vollständigen Menü).
- **Nächste Anweisung**. Bringt den Cursor zur nächsten auszuführenden Anweisung (nur im vollständigen Menü).
- **Ausgabebildschirm**. Schaltet zwischen der Bildschirmanzeige des QuickBASIC-Editors und der Programmausgabe um.
- **Bearbeiten Include-Datei**. Lädt Include-Dateien, auf die Bezug genommen wurde, zum Einsehen und Bearbeiten (nur im vollständigen Menü).
- **Anzeigen Include-Datei**. Zeigt den Inhalt der Include-Dateien, auf die Bezug genommen wurde. Die Include-Datei kann jedoch nicht bearbeitet werden.

14.1 Der Befehl SUBs

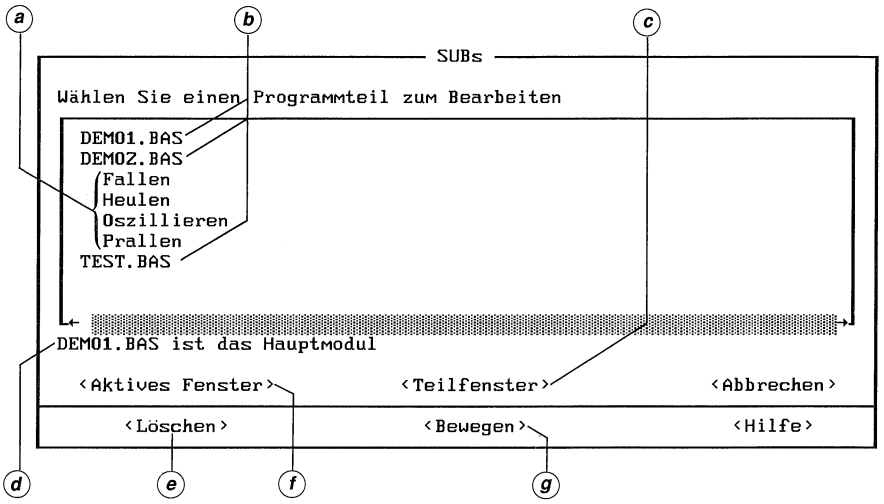


Der Befehl **SUBs** bietet dem Anwender die Wahl zwischen den aktuell geladenen Dateien, deren **SUB**- und **FUNCTION**-Prozeduren, Include-Dateien und Dokumenten. Die ausgewählte Datei oder Prozedur erscheint anschließend im Arbeitsfenster.

Das Dialogfeld **SUBs** steuert sowohl die Datei- als auch die Prozedurverwaltung und kann wie folgt benutzt werden:

1. Wählen Sie den Befehl **SUBs** aus dem Menü **Ansicht**, wobei das in Abbildung 14.1 dargestellte Dialogfeld **SUBs** erscheint.

Abbildung 14.1 Das Dialogfeld SUBs



a) Prozeduren werden unter den entsprechenden Modulen eingerückt.

b) Module werden in Großbuchstaben aufgelistet.

c) Behält das aktive Fenster bei und öffnet ein neues Fenster, indem das ausgewählte Element erscheint.

d) Beschreibt das im hervorgehobenen Feld vorhandene Element.

e) Entfernt ein Modul aus dem Arbeitsspeicher oder löscht eine SUB- oder FUNCTION-Prozedur endgültig aus einem Modul.

f) Bringt eine Prozedur ins aktive Fenster.

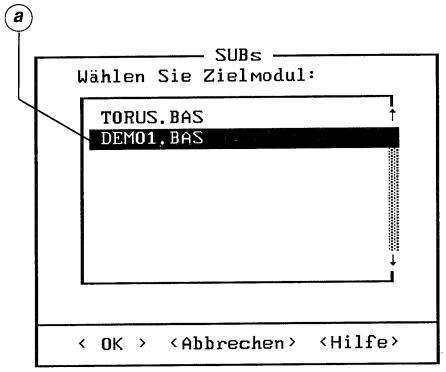
g) Schiebt Prozeduren von einem Modul in ein anderes.

2. Heben Sie das gewünschte Modul, die Prozedur oder Include-Datei durch Betätigen der RICHTUNGSTASTEN hervor. Sie können nun einer der folgenden Aktionen durchführen:

- Wählen Sie die Befehlsfläche **Aktives Fenster** aus, um das hervorgehobene Element ins aktive Fenster zu bringen.
- Wählen Sie die Befehlsfläche **Teilfenster** aus, um das Fenster horizontal aufzuteilen; das untere (aktive) Fenster zeigt die neue Auswahl an, während das obere Fenster den Inhalt des vorhergehenden Fensters enthält. Drücken Sie die Taste F6, um von einem Fenster zum anderen nach unten zu gelangen und die Tastenkombination UMSCHALT+F6, um nach oben zu wechseln. Wählen Sie den Befehl **Teilen** aus dem Menü **Ansicht** erneut aus, um das zusätzliche Fenster zu schließen.

- Wählen Sie die Befehlsfläche **Bewegen**, um die ausgewählte Prozedur in ein anderes Modul zu schieben. Es erscheint das in Abbildung 14.2 gezeigte Dialogfeld, das aus einer Liste der Zielmodule besteht. (Es ist sinnlos, eine Prozedur in einem einmoduligen Programm zu verschieben.)

Abbildung 14.2 Das Dialogfeld Bewegen



a) Aktuelles Zielmodul.

- Wählen Sie die Befehlsfläche **Löschen** aus, um die hervorgehobene Prozedur endgültig aus dem Modul zu entfernen. (Es ist dabei zu beachten, daß das Modul und die dazugehörigen Prozeduren durch den Löschvorgang zwar aus dem Arbeitsspeicher entfernt, jedoch weiterhin auf Diskette/Platte beibehalten werden.)
- Wählen Sie **Abbrechen**, um zum vorhergehenden aktiven Fenster zurückzukehren.
- Wählen Sie **Hilfe**, um nähere Informationen zur Anwendung des Dialogfeldes **SUBs** anzufordern.

Sie sind jetzt in der Lage, die Arbeit mit dem ausgewählten Modul, der Prozedur oder Include-Datei zu beginnen.

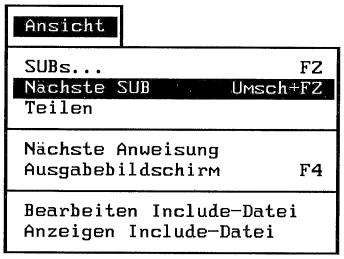
Taste zur direkten Befehlseingabe

F2

Siehe auch

Abschnitt 11.9, "Der Befehl Datei entfernen", Abschnitt 14.2, "Der Befehl Nächste SUB", und Abschnitt 16.7, "Der Befehl Hauptmodul bestimmen".

14.2 Der Befehl Nächste SUB (nur im vollständigen Menü)



Der Befehl **Nächste SUB** zeigt den Inhalt der nächsten Prozedur in alphabetischer Reihenfolge der Namen im aktiven Fenster an. Der Befehl hat jedoch keine Wirkung, wenn das aktuelle Modul keine Prozeduren enthält. Andernfalls können Sie zur nächsten Prozedur weitergehen oder zur vorhergehenden zurückkehren.

Der Befehl **Nächste SUB** ist bei der Arbeit mit Modulen, die nur wenige Prozeduren enthalten, besonders zweckmäßig. In Programmen mit zahlreichen Modulen, können Sie sich mit dem Befehl **SUBs** normalerweise schneller zwischen den einzelnen Teilen des Programms bewegen.

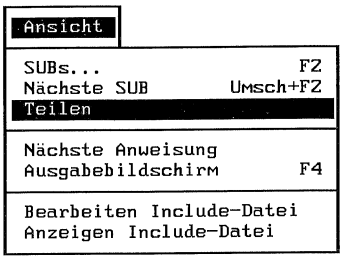
Tastenkombinationen zur direkten Befehlseingabe

UMSCHALT+F2 (vorwärts) und STRG+F2 (rückwärts)

Siehe auch

Abschnitt 14.1, "Der Befehl SUBs".

14.3 Der Befehl Teilen (nur im vollständigen Menü)



Dieser Befehl teilt das Arbeitsfenster horizontal auf und ermöglicht dem Anwender, an zwei Programmteilen gleichzeitig zu arbeiten.

Wenden Sie den Befehl **Teilen** wie folgt an:

1. Wählen Sie den Befehl **Teilen** aus dem Menü **Ansicht**.
2. Drücken Sie F6, um sich von einem Fenster zum anderen nach unten zu bewegen und verschiedene Fenster nacheinander zu aktivieren (oder UMSCHALT+F6 um sich nach oben zu bewegen).
3. Wählen Sie den Befehl **Teilen** erneut aus, um das aktive Fenster auf Gesamtbildschirmgröße zu erweitern und das inaktive Fenster zu schließen.

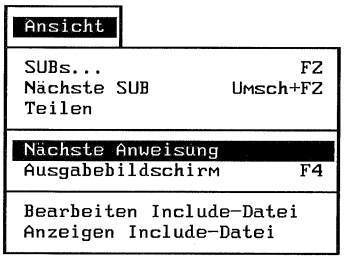
Ein Fenster kann vergrößert, verkleinert oder auf Gesamtbildschirmgröße erweitert werden. Die Fenstergröße läßt sich durch Aktivieren des Fensters und Betätigen einer der folgenden Tastenkombinationen verändern, wobei die zum numerischen Tastenblock gehörenden PLUS- und MINUS-TASTEN zu benutzen sind.

Tastenkombination	*Ergebnis*
ALT+PLUS(+)	Vergrößert das aktive Fenster um eine Zeile.
ALT+MINUS(-)	Verkleinert das aktive Fenster um eine Zeile.
STRG+F10	Erweitert das aktive Fenster auf Gesamtbildschirmgröße. Wenn das bereits der Fall ist, wird die ursprüngliche Größe des Fensters wiederhergestellt.

Siehe auch

Kapitel 10, "Anwendung von QuickBASIC", und Abschnitt 14.1, "Der Befehl SUBs".

14.4 Der Befehl Nächste Anweisung (nur im vollständigen Menü)



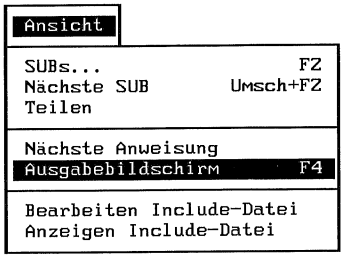
Der Befehl **Nächste Anweisung** bringt den Cursor zur nächsten auszuführenden Anweisung, ohne diese jedoch auszuführen.

Üblicherweise wird der Befehl **Nächste Anweisung** nach Unterbrechung eines Programms beim Debuggen angewandt, da er die Anweisung angibt, die mit Hilfe des Befehls **Weiter** als nächstes ausgeführt wird. Nach Durchgehen eines angehaltenen Programms ist es angebracht, anhand des Befehls **Nächste Anweisung** zum ursprünglichen Punkt der Unterbrechung zurückzukehren.

Siehe auch

Abschnitt 15.2, "Der Befehl Suchen", Abschnitt 16.3, "Der Befehl Weiter", und Kapitel 18, "Das Menü Debug".

14.5 Der Befehl Ausgabebildschirm



Der Befehl **Ausgabebildschirm** veranlaßt QuickBASIC zwischen der Programmierumgebung und der vom Programm erstellten Ausgabe zu wechseln.

Der Befehl ist besonders nützlich, nachdem ein Programm Daten auf den Bildschirm ausgegeben hat. Nach Ausführung eines Programms mit Bildschirmausgabe gibt QuickBASIC am Ende des Programms die Meldung

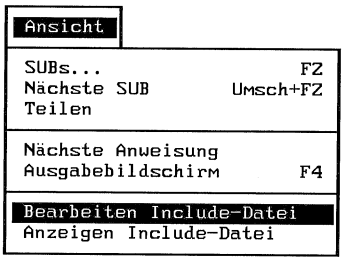
```
Beliebige Taste zum Fortsetzen drücken
```

am unteren Ende des Ausgabebildschirms aus. Nach der Rückkehr zum QuickBASIC-Editor können Sie zwischen Editor und dem Ausgabebildschirm mit Hilfe des Befehls **Ausgabebildschirm** wechseln. Auf diese Weise können Sie sich während der Bearbeitung und Fehlerbeseitigung im Programm auf die vorhergehende Ausgabe beziehen.

Taste zur direkten Befehlseingabe

F4

14.6 Der Befehl Bearbeiten Include-Datei (nur im vollständigen Menü)



Dieser Befehl lädt eine Datei, auf die mit Hilfe des Metabefehls **$INCLUDE** Bezug genommen wird, zum Bearbeiten.

Wählen Sie den Befehl **Bearbeiten Include-Datei** während sich der Cursor in der Zeile mit dem Metabfehl **$INCLUDE** befindet, um den Text der Include-Datei zur Bearbeitung ins aktive Fenster zu laden. QuickBASIC trennt den Text von Include-Dateien in gleicher Weise wie den Text der **SUB**- oder **FUNCTION**-Prozeduren vom restlichen Programmtext. Darüber hinaus behandelt QuickBASIC Include-Dateien getrennt vom Rest des Programms.

Falls Sie eine Include-Datei bearbeiten möchten, ohne den Cursor von der Programmzeile mit dem **$INCLUDE**-Metabefehl zu entfernen, verfahren Sie wie folgt:

1. Teilen Sie den Bildschirm mit Hilfe des Befehls **Teilen** aus dem Menü **Ansicht** auf.
2. Wählen Sie den Befehl **Lade Datei** aus dem Menü **Datei**, um die betreffende Include-Datei zu laden, und vergewissern Sie sich, daß die Option **Include** ausgewählt ist.
3. Bearbeiten Sie die Include-Datei.
4. Wählen Sie den Befehl **Speichern** aus dem Menü **Datei** aus, um die an der Include-Datei vorgenommenen Änderungen abzuspeichern.
5. Kehren Sie durch Betätigen der Taste F6 zu dem im anderen Arbeitsfenster vorhandenen Programm zurück.
6. Schließen Sie das zusätzliche Fenster, indem Sie den Befehl **Teilen** aus dem Menü **Ansicht** erneut ausführen.

Speichern Sie die Include-Datei, um die Änderungen einzugliedern. Sollten Sie dies vergessen, fordert Sie QuickBASIC dazu auf.

14.6.1 Verschachteln von Include-Dateien

Während ein Programm eine beliebige Anzahl von **$INCLUDE**-Metabefehlen enthalten kann, ist das "Verschachteln" von Include-Dateien beschränkt.

Verschachteln bezeichnet in diesem Fall das Auftreten von **$INCLUDE**-Metabefehlen innerhalb einer anderen Include-Datei. Include-Dateien können über höchstens 5 Ebenen verschachtelt werden. Wenn ein selbstbezogener Verweis (in der Form eines **$INCLUDE**-Metabefehls, der sich auf sich selbst oder auf eine Datei bezieht, die ihrerseits auf diesen Metabefehl Bezug nimmt) in einer Include-Datei auftritt, erscheint die Fehlermeldung `Zu viele Dateien` beim Versuch, das Programm auszuführen.

14.6.2 Auffinden von Include-Dateien

Beim Verarbeiten eines **$INCLUDE**-Metabefehls sucht QuickBASIC nach der entsprechenden Include-Datei zuerst in allen im Metabefehl angegebenen Verzeichnissen, dann im Arbeitsverzeichnis und schließlich in allen vom Befehl **Suchpfade festlegen** angegebenen Verzeichnissen für Include-Dateien. Bei der Bezugnahme auf ein Verzeichnis im **$INCLUDE**-Metabefehl selbst, ist ein vollständiger oder relativer Pfadname anzugeben. Folgende Zeile weist beispielsweise QuickBASIC an, nach der Include-Datei mit dem Namen `deklar.bi` im Verzeichnis `c:\include` zu suchen:

```
' $INCLUDE: 'C:\include\deklar.bi'
```

Dies kann auch in Form eines relativen Pfades, wie im folgenden, angegeben werden:

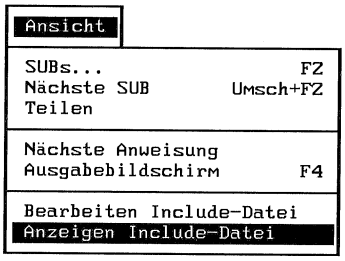
```
' $INCLUDE: '..\include\deklar.bi'
```

Obwohl jegliche Erweiterung, einschließlich *.bas* für Dateien zulässig ist, die mit Hilfe des Metabefehls **$INCLUDE** im Programm eingegliedert werden, empfiehlt es sich, eine eindeutige Erweiterung (wie *.bi*) für Include-Dateien zu verwenden, um jede Verwechslung mit Modulen des Programms zu vermeiden.

Siehe auch

Abschnitt 14.7, "Der Befehl Anzeigen Include-Datei", und Abschnitt 20.2, "Der Befehl Suchpfade festlegen".

14.7 Der Befehl Anzeigen Include-Datei



Dieser Befehl ermöglicht dem Programmierer, den Inhalt der Include-Datei, auf die Bezug genommen wurde, anzusehen, ohne sie jedoch bearbeiten zu können. Dieser Befehl ist ein "Schalter" und weist einen Punkt auf, wenn er eingeschaltet ist. Erneutes Auswählen schaltet den Befehl aus.

Der Befehl **Anzeigen Include-Datei** läßt den Inhalt der Include-Datei hervorgehoben unter dem entsprechenden **$INCLUDE**-Metabefehl erscheinen. Solange diese Option eingeschaltet ist, können Sie sich nach Auswahl des Befehls zwar innerhalb der Include-Datei bewegen, sie jedoch nicht bearbeiten. Beim Versuch die Include-Datei trotzdem zu bearbeiten, erscheint ein Dialogfeld mit der Frage, ob die Include-Datei bearbeitet werden soll. In diesem Fall lädt QuickBASIC die Datei in den Arbeitsspeicher und zeigt deren Text im aktiven Arbeitsfenster an.

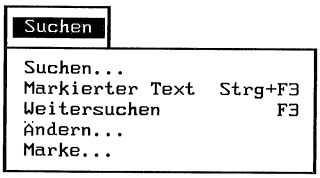
Beim Versuch, eine Programmdatei zu bearbeiten, während der Befehl **Anzeigen Include-Datei** eingeschaltet ist, erscheint ein Dialogfeld mit der Frage, ob dieser Befehl ausgeschaltet werden soll. Wenn Sie die EINGABETASTE betätigen, wird die Include-Datei aus dem aktiven Fenster entfernt, und die Quelldatei kann normal bearbeitet werden.

Verwenden Sie zur Bearbeitung von Dateien, auf die anhand des Metabefehls **$INCLUDE** Bezug genommen wurde, den Befehl **Bearbeiten Include-Datei** oder **Lade Datei** aus dem Menü **Datei**, wobei die Option **Include** auszuwählen ist.

Siehe auch

Abschnitt 14.6, "Der Befehl Bearbeiten Include-Datei".

15 Das Menü Suchen



Das Menü **Suchen** sucht und findet einen "Zieltext" in einem Programm und ersetzt diesen ggf. durch einen neuen Text. Dieses Menü bietet besonders in langen oder komplexen Programmen vielseitige Anwendungsmöglichkeiten. Um sich beispielsweise zu einem entfernten Programmteil zu bewegen, genügt es, nach einer Zeilenmarke zu suchen und auf diese Weise den gewünschten Text in diesem Abschnitt zu finden.

Das Menü **Suchen** umfaßt folgende Befehle:

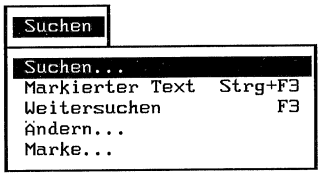
- **Suchen**. Sucht und findet das nächste Vorkommen des gegebenen Suchbegriffs.
- **Markierter Text**. Sucht nach einer Textpassage, die mit dem aktuell hervorgehobenen Text übereinstimmt (nur im vollständigen Menü).
- **Weitersuchen**. Lokalisiert das nächste Vorkommen des im vorhergehenden Suchvorgang angegebenen Texts (nur im vollständigen Menü).
- **Ändern**. Sucht und ersetzt den Zieltext durch einen neuen Text.
- **Marke**. Sucht nach einer gegebenen Zeilenmarke (nur im vollständigen Menü).

15.1 Definieren des Zieltexts

Suchprozeduren erfordern die Angabe eines Zieltexts, nach dem gesucht werden soll. In QuickBASIC gibt es mehrere Methoden zur Angabe des Zieltexts. Das Wort, bei dem sich der Cursor befindet, oder das unmittelbar links vom Cursor steht, wird standardmäßig als Zieltext festgelegt. Befindet sich der Cursor in einer Leerzeile, wird der vorhergehende Suchbegriff zum Standardzieltext. Der Standardzieltext läßt sich durch Eingabe einer neuen Textpassage in den Dialogfeldern **Ändern** oder **Suchen** verändern.

Der Suchvorgang beginnt an der Cursorstelle und verläuft nach unten. Wenn QuickBASIC das Ende des aktiven Fensters erreicht, ohne einen übereinstimmenden Text zu finden, kehrt es an den Anfang der Datei zurück und setzt den Suchvorgang fort, bis der Suchbegriff gefunden oder die ursprüngliche Cursorstelle erreicht wird. Es gibt auch Optionen, die die Suche auf das ganze aktuelle Modul oder auf sämtliche geladene Module erweitern.

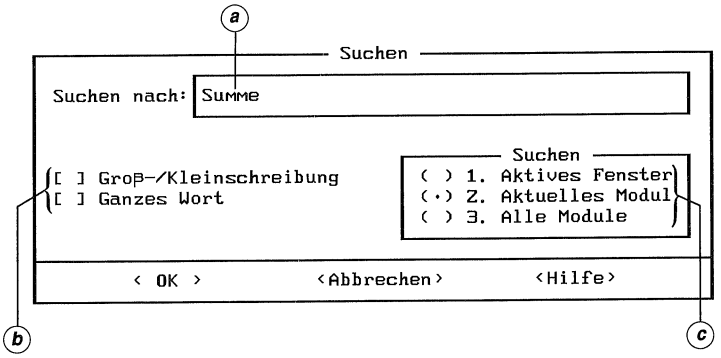
15.2 Der Befehl Suchen



Der Befehl **Suchen** sucht nach einem gegebenen Text und positioniert den Cursor an dieser Stelle. Es kann sich dabei um ein einzelnes Zeichen, ein Wort oder eine beliebige Kombination von Wörtern und Zeichen handeln.

Geben Sie den gewünschten Text im Dialogfeld **Suchen** ein, das in Abbildung 15.1 dargestellt wird.

Abbildung 15.1 Das Dialogfeld Suchen



a) *Suchbegriff (Text).*
b) *Sucheinschränkungen.*
c) *Suchoptionen.*

Das Dialogfeld steuert auch das Ausmaß des Suchvorgangs (der das Arbeitsfenster, das aktuelle Modul, oder alle aktuell geladenen Dateien umfassen kann) sowie alle bei der Suche geltenden Einschränkungen.

Führen Sie einen Suchvorgang wie folgt durch:

1. Wählen Sie den Befehl **Suchen** aus dem Menü **Suchen**. Es erscheint das Dialogfeld **Suchen**.
2. Geben Sie den Suchtext im Textfeld **Suchen nach** ein. (Der Standardzieltext wird hervorgehoben.)
3. Wählen Sie eine Optionsfläche durch Betätigen der TAB-TASTE und der RICHTUNGSTASTEN aus. (Die Optionen sind in Tabelle 15.1 aufgeführt.)
4. Setzen Sie die gewünschten Einschränkungen durch Auswahl der Option mit der TAB-TASTE und Ein- und Ausschalten der Option anhand der LEERTASTE fest. (Die Einschränkungen sind in Tabelle 15.2 aufgelistet.)
5. Starten Sie den Suchvorgang durch Betätigen der EINGABETASTE.

Wenn QuickBASIC den Zieltext nicht findet, gibt es die Meldung `Suchbegriff nicht gefunden` aus und bringt den Cursor zum Startpunkt zurück.

Drücken Sie die EINGABETASTE oder die ESC-TASTE, um die Meldung vom Bildschirm zu löschen und fortzufahren.

Nachstehend werden die in Verbindung mit den Suchbefehlen verfügbaren Optionen aufgelistet:

Tabelle 15.1 Suchoptionen

Option	*Beschreibung*
Aktives Fenster	Sucht nur im aktuellen, aktiven Fenster.
Aktuelles Modul	Sucht nur im aktuellen Modul (einschließlich der Prozeduren).
Alle Module	Sucht in allen Teilen aller geladenen Dateien.

Folgende Tabelle führt die Einschränkungen auf, die anhand des Befehls **Suchen** festgelegt werden können.

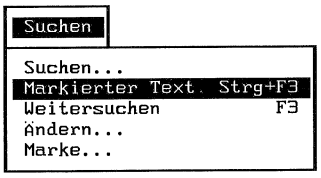
Tabelle 15.2 Sucheinschränkungen

Einschränkung	*Beschreibung*
Groß-/Kleinschreibung	Sucht nach genau übereinstimmenden Textabschnitten, wobei auch die Schreibweise beachtet wird. Wenn beispielsweise `CHR` als Suchbegriff angegeben ist, sucht diese Option nur nach `CHR`, nicht auch nach `Chr`.
Ganzes Wort	Lokalisiert den Suchbegriff nur dann, wenn er als separates Wort vorkommt, das von Leerstellen, Satzzeichen oder anderen nicht zu einem Wort gehörenden Zeichen eingeschlossen ist. Zu den Bestandteilen eines Wortes gehören die Buchstaben von A bis Z und a bis z, die Zahlen von 0 bis 9 und die Zeichen !, #, $, % (die als Suffixe in Vereinbarungen auftreten). Die Suche nach dem ganzen Wort `HEX` würde zum Beispiel `HEX` finden, ohne `HEX$` oder `HEXADEZIMAL` zu berücksichtigen.

Siehe auch

Abschnitt 15.5, "Der Befehl Ändern".

15.3 Der Befehl Markierter Text (nur im vollständigen Menü)



Der Befehl **Markierter Text** aus dem Menü **Suchen** sucht nach Textpassagen, die mit dem im aktiven Fenster ausgewählten Text übereinstimmen.

Dieser Befehl kann mit Hilfe folgender Arbeitsschritte angewandt werden:

1. Markieren Sie den zu suchenden Text, der eine Zeile (256 Zeichen) nicht überschreiten darf.
2. Wählen Sie den Befehl **Markierter Text** aus dem Menü **Suchen** aus.

QuickBASIC hebt das nächste Vorkommen des markierten Texts hervor und positioniert den Cursor auf das erste Zeichen der betreffenden Zeichenkette. Wenn der Suchbegriff nicht im Text auftritt, bleibt der Bildschirm unverändert.

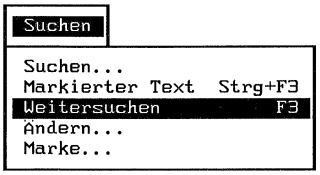
Tastenkombination zur direkten Befehlseingabe

STRG+F3

Siehe auch

Abschnitt 15.2, "Der Befehl Suchen".

15.4 Der Befehl Weitersuchen (nur im vollständigen Menü)



Der Befehl **Weitersuchen** sucht nach dem nächsten Vorkommen des vorhergehenden Suchbegriffs. Der im vorhergehenden Befehl **Suchen** oder **Ändern** angegebene Suchbegriff wird zum Zieltext des durch **Weitersuchen** veranlaßten Suchvorgangs.

Falls jedoch der Befehl **Weitersuchen** ausgewählt wird, ohne vorher die Befehle **Suchen** oder **Ändern** angewandt zu haben, nimmt QuickBASIC das Wort, bei dem sich der Cursor befindet, oder das unmittelbar links vom Cursor steht, als Standardzieltext an.

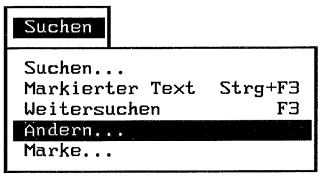
Taste zur direkten Befehlseingabe

F3

Siehe auch

Abschnitt 15.2, "Der Befehl Suchen".

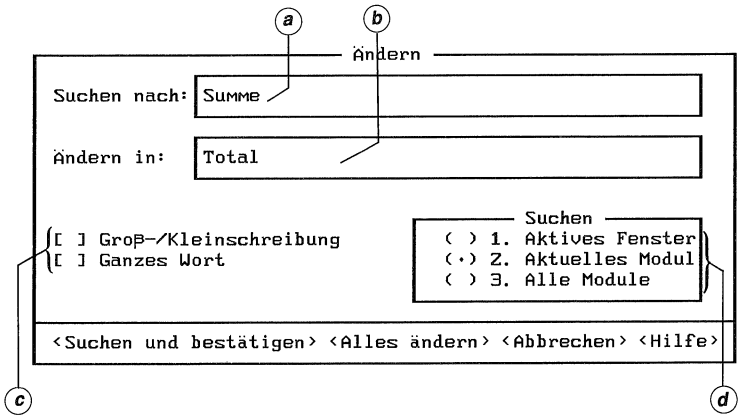
15.5 Der Befehl Ändern



Der Befehl **Ändern** sucht nach einer Zeichenkette, um sie durch eine andere Zeichenkette zu ersetzen. Die Zeichenketten können aus jeweils einem Zeichen, einem Wort oder einer beliebigen Wort- und Zeichenkombination bestehen.

Das in Abbildung 15.2 dargestellte Dialogfeld **Ändern** dient der Eingabe des Ziel- und Ersatztexts sowie der Festlegung der Suchattribute und -einschränkungen.

Abbildung 15.2 Das Dialogfeld Ändern



a) Zieltext.
b) Ersatztext.
c) Sucheinschränkungen.
d) Suchoptionen.

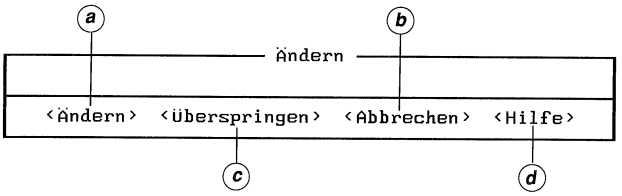
Text läßt sich wie folgt ändern:

1. Wählen Sie den Befehl **Ändern** aus dem Menü **Suchen**.
2. Geben Sie den zu ersetzenden Text im Textfeld **Suchen nach** ein.
3. Geben Sie den Ersatztext im Textfeld **Ändern in** ein.
4. Wählen Sie die gewünschten Optionen aus (siehe Tabelle 15.1).
5. Legen Sie die Sucheinschränkungen fest (siehe Tabelle 15.2).

6. Wählen Sie den Befehl **Suchen** oder **Suchen und Bestätigen**, um mit dem Ersetzen des Texts zu beginnen, oder **Abbrechen**, um den Befehl abzubrechen.

   - Der Befehl **Suchen und Bestätigen** findet den Zieltext und läßt anschließend vier weitere Befehlsflächen auf dem Bildschirm erscheinen: **Ändern**, **Überspringen**, **Abbrechen** und **Hilfe**. Mit Hilfe der ersten wird die Änderung durchgeführt, während die zweite nach dem nächsten Vorkommen sucht (ohne die Änderung vorzunehmen). **Abbrechen** bricht den Suchvorgang ab und kehrt zum Ausgangspunkt zurück. Durch Auswählen der Befehlsfläche **Hilfe** erhalten Sie weitere Informationen zum Dialogfeld **Suchen und Bestätigen** (siehe Abbildung 15.3).

Abbildung 15.3 Die Dialogfelder Ändern, Überspringen, Abbrechen und Hilfe



a) Führt die Änderung aus.
b) Bricht den Suchvorgang ab und kehrt zum aktiven Fenster zurück.
c) Sucht nach dem nächsten Vorkommen.
d) Bietet Hilfe für das Feld Ändern.

- Der Befehl **Alles ändern** sucht und ersetzt alle Vorkommen des Zieltexts, ohne eine Bestätigung abzuwarten.
- Durch **Abbrechen** wird die Kontrolle an das aktive Fenster zurückgegeben.
- Wird der Zieltext nicht gefunden, zeigt QuickBASIC die Meldung

  ```
  Suchbegriff nicht gefunden
  ```

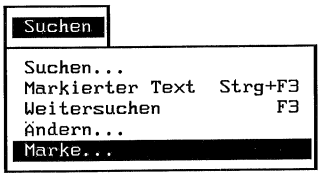
  an. Löschen Sie diese Meldung mit ESC, und fahren Sie fort.

Wichtig Der Befehl **Rückgängig machen** aus dem Menü **Bearbeiten** kann keine mit Hilfe des Befehls **Ändern** vorgenommene Änderung rückgängig machen. Bei der Anwendung des Befehls **Alles ändern** ist Vorsicht angebracht, da er das Programm durchgreifend verändern kann.

Siehe auch

Abschnitt 15.2, "Der Befehl Suchen"

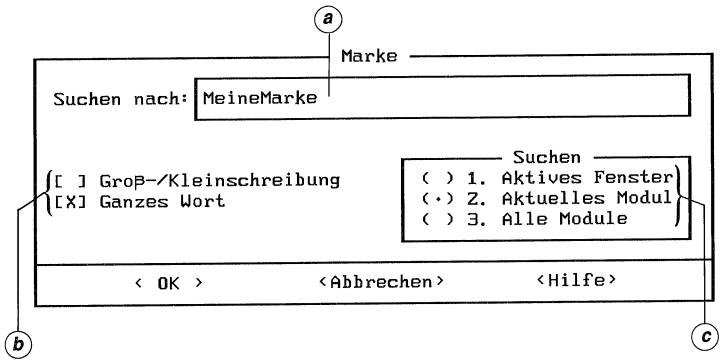
15.6 Der Befehl Marke (nur im vollständigen Menü)



Der Befehl **Marke** sucht nach einer Zeilenmarke. Da letztere durch einen Doppelpunkt abgeschlossen wird, sucht QuickBASIC nach dem jeweiligen Eintrag im Textfeld und nach einem Doppelpunkt. Bei Eingabe des Zieltexts `MeineMarke` im Dialogfeld **Marke** sucht QuickBASIC nach `MeineMarke:`

Das entsprechende Dialogfeld **Marke** wird in Abbildung 15.4 veranschaulicht.

Abbildung 15.4 Das Dialogfeld Marke

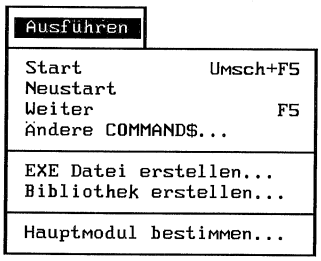


a) Name der gesuchten Zeilenmarke.
b) Scheinschränkungen.
c) Suchoptionen.

Siehe auch

Abschnitt 15.2, "Der Befehl Suchen", und Abschnitt 15.3, "Der Befehl Markierter Text".

16 Das Menü Ausführen



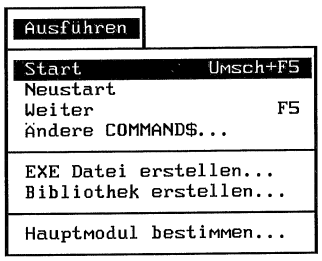
Die zum Menü **Ausführen** gehörenden Befehle erlauben die Kontrolle des Programmablaufs innerhalb von QuickBASIC, die Kompilierung selbständiger Programme mit Hilfe des BC-Compilers und die Erstellung von Quick-Bibliotheken. Dadurch sind Sie in der Lage, ein Programm innerhalb der QuickBASIC-Umgebung zu erstellen und zu debuggen sowie die Ausführung des Programms durch Kompilieren zu beschleunigen. Die Programme können bei selbständigen Anwendungen auch direkt von der DOS-Befehlszeile ausgeführt werden, ohne vorher QuickBASIC zu laden. Die Hauptmodule eines Programms werden ebenfalls anhand des Menüs **Ausführen** definiert.

Das Menü **Ausführen** verfügt über folgende Befehle:

- **Start**. Führt das aktuell geladene Programm aus der QuickBASIC-Umgebung aus.
- **Neustart**. Setzt alle Variablen auf Null und alle Zeichenkettenausdrücke auf die Länge Null zurück und bereitet die Ausführung der ersten Anweisung im Programm vor.
- **Weiter**. Setzt die Ausführung eines unterbrochenen Programms mit der Anweisung fort, an der die Ausführung angehalten wurde (ohne Variablen oder Ausdrücke zurückzusetzen).
- **Ändere COMMAND$**. Greift auf eine Zeichenkette zu, die dem Programm mit Hilfe der Funktion **COMMAND$** aus der DOS-Befehlszeile übergeben wurde (nur im vollständigen Menü verfügbar).

- **EXE-Datei erstellen.** Erstellt eine ausführbare Datei aus dem aktuell geladenen Programm.
- **Bibliothek erstellen.** Legt eine Quickbibliothek an (nur im vollständigen Menü).
- **Hauptmodul bestimmen.** Legt ein ausgewähltes Modul als Hauptmodul fest (nur im vollständigen Menü).

16.1 Der Befehl Start



Der zum Menü **Ausführen** gehörende Befehl **Start** führt das aktuell geladene Programm aus.

Ein Programm kann nach dem Start durch Betätigen der Tastenkombination STRG+UNTBR angehalten und mit Hilfe des Befehls **Weiter** aus dem Menü **Ausführen** von dem Punkt der Unterbrechung fortgesetzt werden. Der Befehl **Start** führt das gesamte Programm erneut aus.

Hinweis Die in Anhang A, "Übertragen von BASICA-Programmen in QuickBASIC", des Handbuches *Programmieren in BASIC* beschriebenen Anweisungen von BASICA-Programmen erfordern Änderungen, um in QuickBASIC richtig ausgeführt zu werden.

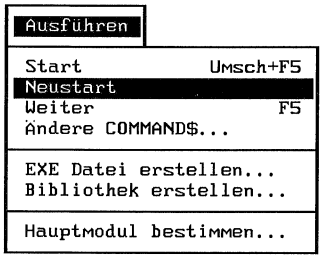
Tastenkombination zur direkten Befehlseingabe

UMSCHALT+F5

Siehe auch

Abschnitt 16.2, "Der Befehl Neustart", Abschnitt 16.3, "Der Befehl Weiter", und Abschnitt 16.5, "Der Befehl EXE-Datei erstellen".

16.2 Der Befehl Neustart



Der Befehl **Neustart** aus dem Menü **Ausführen** setzt alle Variablen auf den Wert Null und sämtliche Zeichenkettenausdrücke auf die Länge Null zurück. Der Befehl bereitet das Programm für die Ausführung vor, ohne dieses jedoch zu starten, und hebt die erste ausführbare Anweisung des Programms hervor.

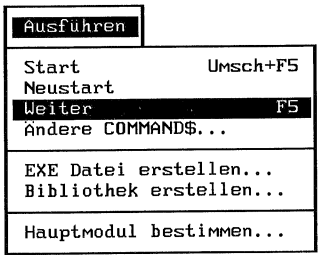
Im allgemeinen wird der Befehl **Neustart** während der Fehlerbeseitigung eingesetzt, wenn die Ausführung nicht fortgesetzt werden soll. Er erweist sich besonders dann als nützlich, wenn Sie das Programm neu starten und den vor dem ersten Haltepunkt oder der ersten Stoppbedingung stehenden Programmcode mit Hilfe der Taste F8 im Einzelschrittmodus durchgehen wollen. (Die in Abschnitt 18.3, "Der Befehl Stoppbedingung", und Abschnitt 18.8, "Der Befehl Haltepunkt ein/aus" erläuterten Haltepunkte und Stoppbedingungen halten die Programmausführung an.) Möchten Sie dagegen das Programm bis zum ersten Haltepunkt oder zur ersten Stoppbedingung ausführen, ohne den Code schrittweise durchzugehen, ist der Befehl **Start** anzuwenden.

Mit Hilfe des Befehls **Weiter** aus dem Menü **Ausführen** oder des Befehls **Nächste Anweisung festlegen** aus dem Menü **Debug**, gefolgt von dem Befehl **Weiter**, kann das Programm von einer bestimmten Stelle ausgeführt werden, ohne die Variablen und Ausdrücke zurückzusetzen.

Siehe auch

Abschnitt 16.1, "Der Befehl Start", Abschnitt 16.3, "Der Befehl Weiter", und Abschnitt 18.11, "Der Befehl Nächste Anweisung festlegen".

16.3 Der Befehl Weiter



Der Befehl **Weiter** setzt ein unterbrochenes Programm von der letzten ausgeführten Anweisung fort oder startet es neu.

Der Befehl **Weiter** wird hauptsächlich während der Fehlerbeseitigung verwendet, um das Programm von einem Haltpunkt oder einer Stoppbedingung zum/zur anderen auszuführen. Die Unterbrechung der Ausführung erlaubt Ihnen, den Code zu analysieren und die erforderlichen Änderungen vorzunehmen. Der Programmablauf wird anschließend anhand des Befehls **Weiter** fortgesetzt. Wenn die jeweilige Änderung die Fortsetzung des Programms von der aktuellen Anweisung verhindert, hält QuickBASIC an, um zu fragen, ob die Ausführung ohne die Änderung fortgesetzt oder das Programm unter Eingliederung der Änderung neu gestartet werden soll. (Falls Sie das Programm nicht erneut starten möchten, haben Sie die Möglichkeit, das Direktfenster zum Ausführen von zeitweiligen Änderungen zu verwenden. Dies erlaubt die Fortsetzung des Programms.)

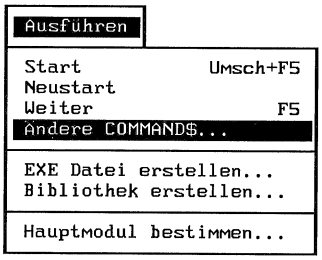
Taste zur direkten Befehlseingabe

F5

Siehe auch

Abschnitt 16.1, "Der Befehl Start", und Abschnitt 16.2, "Der Befehl Neustart".

16.4 Der Befehl Ändere COMMAND$ (nur im vollständigen Menü)



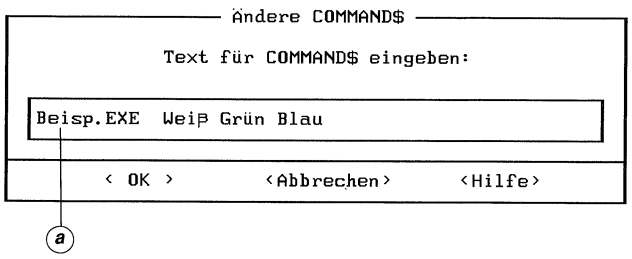
Der Befehl **Ändere COMMAND$** dient fortgeschrittenen Programmierern beim Zugriff auf Textzeichenketten, die dem jeweiligen Programm aus der DOS-Befehlszeile durch die Funktion **COMMAND$** übergeben wurden. Demnach ist der Befehl nur dann zweckmäßig, wenn das Programm die Funktion **COMMAND$** verwendet.

In einigen Fällen ist die Übergabe von Informationen aus der DOS-Umgebung an das Anwendungsprogramm nötig. Die Eingabe von `QB DEMO1` teilt QuickBASIC beispielsweise mit, daß der Programmierer mit dem Programm `DEMO1` innerhalb der QuickBASIC-Umgebung arbeiten möchte. In diesem Fall bildet `DEMO1` die an QuickBASIC aus der Befehlszeile übergebene Information.

Einem Programm können Informationen aus der DOS-Befehlszeile mit Hilfe der Funktion **COMMAND$** übergeben werden. Der Arbeitsspeicher wird bei jedem Programmstart gelöscht, wobei die Information aus der DOS-Befehlszeile verloren geht. Während der Entwicklung und Fehlerbeseitigung empfiehlt es sich, das Programm zu testen, ohne bei jeder Programmänderung zu DOS zurückkehren zu müssen. Der Befehl **Ändere COMMAND$** weist QuickBASIC an, die von der Funktion **COMMAND$** zurückgegebene Zeichenkette durch den im Dialogfeld eingegebenen Text zu ersetzen.

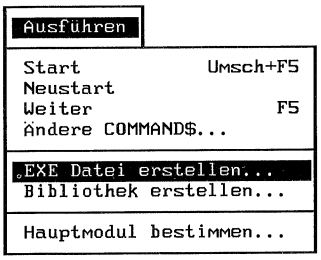
Im Dialogfeld **Ändere COMMAND$** wird die von der Funktion **COMMAND$** zurückgegebene Zeichenkette angegeben (siehe Abbildung 16.1).

Abbildung 16.1 Das Dialogfeld Ändere COMMAND$



a) *Der von der Funktion COMMAND$ zurückgegebene Text.*

16.5 Der Befehl EXE-Datei erstellen



Nachdem Ihr Programm in der QuickBASIC-Umgebung lauffähig ist, können Sie mit Hilfe des Befehls **EXE-Datei erstellen** eine Programmversion anlegen, die direkt aus der DOS-Befehlszeile ausführbar ist, ohne zuvor in QuickBASIC geladen zu werden. Die zu diesem Zweck im Programm erforderliche Datei wird als ausführbare Datei bezeichnet und ist mit der Erweiterung *.exe* versehen. Das Programm wird durch Eingabe des Basisnamens der ausführbaren Datei (ohne der Erweiterung) nach der DOS-Eingabeaufforderung ausgeführt.

Beim Erstellen einer ausführbaren Datei verwendet QuickBASIC zuerst den BASIC-Befehlszeilencompiler (BC) zum Kompilieren des BASIC-Quellcodes zu einer intermediären Datei (Objektdatei). Anschließend bindet QuickBASIC sämtliche getrennt kompilierten Module des Programms in einer einzigen ausführbaren Datei mit Hilfe des als Microsoft Overlay Linker (LINK) bezeichneten Programms. LINK kombiniert ebenfalls die kompilierten von BC erstellten Objektdateien mit den Routinen, die einen korrekten Programmablauf gewährleisten und sich in den Laufzeitbibliotheken *brun45.lib* und *bcom45.lib* befinden.

16.5.1 Die Erstellung ausführbarer Dateien

Die in nachstehender Liste aufgeführten Dateien müssen QuickBASIC bei der Erstellung ausführbarer Dateien zur Verfügung stehen. Vergewissern Sie sich, daß sich diese im aktuellen Verzeichnis befinden oder durch die Umgebungsvariablen PATH- oder LIB verfügbar sind. Sie können QuickBASIC die Position dieser Dateien auch anhand der Option **Suchpfade festlegen** mitteilen. (Weitere Information über das Festlegen von Umgebungsvariablen sind Abschnitt 20.2, "Der Befehl Suchpfade festlegen", zu entnehmen).

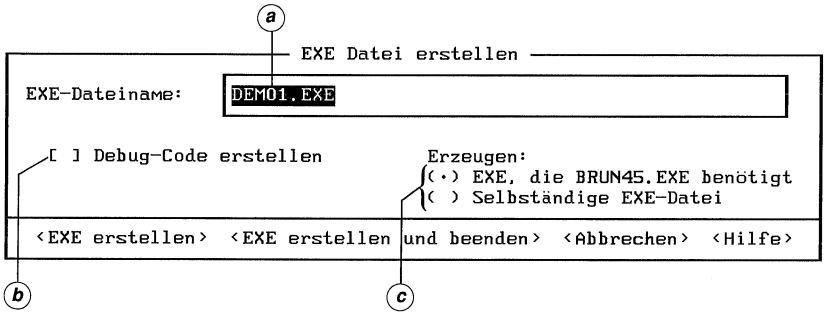
Programm	***Funktion***
bc.exe	Erstellt aus dem Programm Objektdateien, die mit der Erweiterung *.obj* versehen sind und mit Hilfe von LINK gebunden werden können.
link.exe	Bindet die mit BC angelegten Objektdateien.

Programm	*Funktion*
brun45.lib (nur im vollständigen Menü)	Stellt unterstützende Routinen zur Verfügung, die einem BASIC-Programm die Durchführung von Aufgaben wie Annahme der Eingabe oder Textausgabe auf dem Bildschirm, erlauben. Diese Routinen werden durch *link.exe* in das Programm eingebunden. Die Bibliothek *brun45.lib* bildet die Laufzeitbibliothek des Moduls. Die mit dieser Bibliothek gebundenen Programme bestehen zwar aus kleinen ausführbaren Dateien, können jedoch nur in Verbindung mit dem Laufzeitmodul *brun45.exe* ausgeführt werden.
bcom45.lib	Liefert genau wie *brun45.exe* unterstützende Routinen, mit dem Unterschied, daß die anhand dieser Bibliothek erstellten ausführbaren Dateien keine Unterstützung durch das Laufzeitmodul *brun45.exe* erfordern. Die mit *bcom45.lib* angelegten Dateien sind zwar umfangreicher als die mit *brun45.exe* erstellten, lassen sich jedoch als selbständige Programme ausführen.

Verfahren Sie wie folgt, um eine ausführbare Datei aus einem in QuickBASIC geladenen Programm anzulegen:

1. Wählen Sie den zum Menü **Ausführen** gehörenden Befehl **EXE-Datei erstellen**, um das Dialogfeld **EXE-Datei erstellen** erscheinen zu lassen (siehe Abbildung 16.2).

Abbildung 16.2 Das Dialogfeld EXE-Datei erstellen



a) Name der ausführbaren Datei.
b) Bietet auf Anforderung zusätzliche Informationen zum Debuggen.
c) Optionen zur Erstellung der ausführbaren Datei.

2. Falls Sie die Datei umbenennen möchten, geben Sie den gewünschten Basisnamen im Textfeld ein. Andernfalls brauchen Sie im Textfeld keine Änderung vorzunehmen.

3. Wählen Sie die Option **EXE, die BRUN45.EXE benötigt** oder die Option **Selbständige EXE-Datei** aus.

   Diese Alternativen werden in Abschnitt 16.5.3, "Ausführbare Dateitypen", eingehend erläutert.

4. Wählen Sie die Option **Debug-Code erstellen** aus, wenn Sie die jeweilige ausführbare Datei mit zusätzlicher Fehlerprüfung versehen möchten. In diesem Fall wird die Position der Fehler während der Laufzeit angegeben. Wenn diese Option aktiviert wird, erkennt das Programm auch die Tastenkombination STRG+UNTBR an. Es ist jedoch zu beachten, daß die mit dieser Option erstellten Dateien zu CodeView® nicht kompatibel sind und ein umfangreicheres und langsameres Programm bilden. Die Beschreibung der von dieser Option behandelten Fehler ist in Abschnitt 16.5.4, "Fehlerprüfung während der Laufzeit ausführbarer Dateien", zu finden.

5. Wählen Sie die Befehlsfläche **EXE erstellen** oder **EXE erstellen und beenden**:

   - Die Befehlsfläche **EXE erstellen** legt die ausführbare Datei an und kehrt dann zu QuickBASIC zurück.
   - Die Befehlsfläche **EXE erstellen und beenden** legt die ausführbare Datei an und beendet QuickBASIC.

16.5.2 Quick-Bibliotheken und ausführbare Dateien

Verwendet das Programm Routinen aus einer Quick-Bibliothek, muß die betreffende Quick-Bibliothek gleichzeitig mit dem Programm geladen werden. Wenn eine Quick-Bibliothek bei Auswahl des Befehls **EXE-Datei erstellen** geladen ist, sucht der BC-Compiler beim Erstellen der ausführbaren Datei nach einer selbständigen Bibliothek (*.lib*), die der geladenen Quick-Bibliothek entspricht.

Quick-Bibliotheken ermöglichen einen schnellen Zugriff auf eine eigene Prozedurenbibliothek, können jedoch nur innerhalb der QuickBASIC-Umgebung eingesetzt werden. Der Compiler muß auf die entsprechende selbständige Bibliothek (*.lib*) zugreifen können, um die in der Quick-Bibliothek enthaltenen Prozeduren in die ausführbaren Datei einzuschließen. Aus diesem Grund ist es empfehlenswert, daß Quick- und .LIB-Bibliotheken in einem einzigen Verzeichnis enthalten sind. Der Pfad zu den beim Erstellen einer ausführbaren Datei erforderlichen .LIB- oder Quick-Bibliotheken kann im Befehl **Suchpfade festlegen** aus dem Menü **Optionen** angegeben werden. (Der Befehl **Bibliothek erstellen** aus dem Menü **Ausführen** erstellt automatisch beide Bibliothektypen. Näheres über die Erstellung und Verwendung von Quick-Bibliotheken entnehmen Sie bitte Abschnitt 16.6, "Der Befehl Bibliothek erstellen", aus diesem Handbuch und dem zu *Programmieren in BASIC* gehörenden Anhang H.)

16.5.3 Ausführbare Dateitypen

Sie haben die Möglichkeit, zwei verschiedene Dateitypen zu erstellen. Die Entscheidung hängt davon ab, ob das jeweilig ausführbare Programm auf das Laufzeitmodul *brun45.exe* zugreifen soll oder nicht.

16.5.3.1 Programme mit Zugriff auf das Laufzeitmodul

Die mit der Option **EXE, die BRUN45.EXE benötigt** kompilierten Programme greifen während der Ausführung auf das Laufzeitmodul *brun45.exe* zu, da dieses den zur Implementierung der BASIC-Sprache erforderlichen Code enthält. Laufzeitmodule bieten folgende Vorteile im Vergleich zu selbständigen Programmen:

- Die ausführbare Datei ist viel kleiner.
- **COMMON**-Variablen und geöffnete Dateien bleiben über **CHAIN**-Anweisungen hinweg erhalten und erlauben den Programmen die gemeinsame Nutzung der Daten. In einem selbständigen Programm bleiben geöffnete Dateien oder in **COMMON**-Anweisungen aufgeführte Variablen bei der Übergabe der Kontrolle an ein anderes Programm anhand der **CHAIN**-Anweisung nicht erhalten.
- Da das Laufzeitmodul *brun45.exe* im Arbeitsspeicher verbleibt, braucht es in einer Folge von verketteten Programmen nicht für jedes einzelne Programm erneut geladen zu werden.

16.5.3.2 Selbständige Programme

Die mit der Option **Selbständige EXE-Datei** erstellten ausführbaren Dateien erfordern keinen Zugriff auf das Laufzeitmodul *brun45.exe*. Da diese Dateien jedoch die gleichen Unterstützungsroutinen wie *brun45.exe* umfassen, sind sie umfangreicher als die mit der Option **EXE, die BRUN45.EXE benötigt** angelegten Dateien. In mit der Option **Selbständige EXE-Datei** erstellten Dateien bleiben in **COMMON**-Anweisungen aufgeführte Dateien oder Variablen nicht erhalten, wenn die Kontrolle anhand einer **CHAIN**-Anweisung an ein anderes Programm übergeben wird.

Selbständige Dateien weisen folgende Vorteile auf:

- Etwas schnellere Ausführung als Laufzeitprogramme.
- Einsparen von RAM-Speicherplatz im Fall kleiner, einfacher Programme, die nicht alle Routinen des Laufzeitmoduls beanspruchen.
- Das Programm erfordert kein Laufzeitmodul und kann mit jedem DOS-Computer selbständig ausgeführt werden.

16.5.4 Fehlerprüfung während der Laufzeit ausführbarer Dateien

Wenn Sie beim Erstellen einer ausführbaren Datei das Kontrollfeld **Debug-Code erstellen** einschalten, wird das Programm während der Ausführung auf folgendes geprüft:

- Arithmetischer Überlauf. Sämtliche arithmetische Operationen sowohl Ganzzahl- als auch Gleitkommaoperationen werden auf Über- und Unterlauf geprüft.
- Datenfeldgrenzen. Die Datenfeldindizes werden überprüft, um sicherzustellen, daß sie im Bereich der Angaben in den **DIM**-Anweisungen liegen.
- Zeilenpositionen. Der erzeugte Binärcode verfügt über zusätzliche Tabellen, so daß die Laufzeitfehlermeldungen die Zeilen, in denen Fehler auftreten, angeben.
- **RETURN**-Anweisungen. Alle **RETURN**-Anweisungen werden auf das Vorhandensein einer vorhergehenden **GOSUB**-Anweisung geprüft.
- Die Tastenkombination STRG+UNTBR, die die Programmausführung anhält. Während des Programmablaufs prüft das Programm nach Ausführung jeder Zeile, ob der Benutzer die Tastenkombination STRG+UNTBR inzwischen betätigt hat. In diesem Fall wird das Programm angehalten.

  Bitte beachten Sie, daß die Option **Debug-Code erstellen** beim Kompilieren eines Programms eingeschaltet sein muß, wenn das Programm die Tastenkombination STRG+UNTBR erkennen soll. Andernfalls wird das Programm bei Betätigen der Tastenkombination STRG+UNTBR nur unter folgenden Voraussetzungen angehalten:

  1. Ein Benutzer gibt Daten als Antwort auf die Eingabeaufforderung einer **INPUT**-Anweisung ein.
  2. Der Programmcode prüft ausdrücklich auf die Betätigung der Tastenkombination STRG+UNTBR.

Warnung Wenn die Option **Debug-Code erstellen** nicht verwendet wird, veranlassen Fehler der Datenfelddimensionierung, **RETURN**-Anweisungen ohne entsprechende **GOSUB**-Anweisungen, oder arithmetischer Überlauf nicht die Ausgabe einer Fehlermeldung und das Verhalten des Programms wird daher unvorhersehbar.

16.5.5 Gleitkommaarithmetik in ausführbaren Dateien

Ein Programm, das zu einer ausführbaren Datei kompiliert wurde, führt Berechnungen mit Gleitkommavariablen schneller und effizienter durch, als in der QuickBASIC-Umgebung. Ausführbare Dateien optimieren die Geschwindigkeit und Genauigkeit der Ausführung, indem sie die Reihenfolge der arithmetischen Operationen verändern. (Der numerische Koprozessor bzw. die Emulation der Koprozessorfunktionen werden besser ausgenutzt.)

Diese zusätzliche Genauigkeit läßt sich anhand der Ergebnisse der Anwendung von Vergleichsoperatoren mit Variablen einfacher und doppelter Genauigkeit feststellen. Der folgende Codeabschnitt gibt beispielsweise beim Programmablauf in der QuickBASIC-Umgebung das Ergebnis `Gleich` aus, während er bei der Ausführung der entsprechenden ausführbaren Datei das Ergebnis `Nicht gleich` ausgibt:

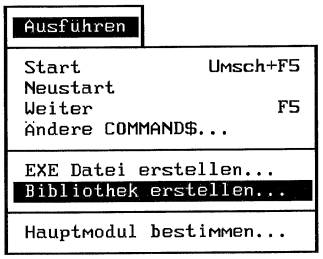
```
B! = 1.0
A! = B!/3.0
.
.
.
IF A!=B!/3.0 THEN PRINT "Gleich" ELSE PRINT "Nicht gleich"
```

Das Problem ergibt sich aus der Durchführung der Berechnung `B!/3.0` innerhalb eines Vergleichs. Das kompilierte Programm speichert das Ergebnis dieser Berechnung im mathematischen Koprozessor-Chip ab, der dem Wert eine höhere Genauigkeit verleiht als dem des in der Variablen `A!` gespeicherten Wertes, und somit die Ungleichheit verursacht.

Die in Vergleichen auftretenden Probleme können durch Durchführung aller Berechnungen außerhalb von Vergleichen vermieden werden. Der folgende umgeschriebene Codeabschnitt führt sowohl beim Ablauf in der QuickBASIC-Umgebung als auch bei der Ausführung der ausführbaren Datei zu dem gleichen Ergebnis:

```
B!=1.0
A!=B!/3.0
.
.
.
Tmp!=B!/3.0
IF A!=Tmp! THEN PRINT "Gleich" ELSE PRINT "Nicht gleich"
```

16.6 Der Befehl Bibliothek erstellen (nur im vollständigen Menü)



Der Befehl **Bibliothek erstellen** legt eine Quick-Bibliothek für kompilierte Programme an. Eine Quick-Bibliothek kann Prozeduren enthalten, die in mehreren Programmen zur Anwendung kommen.

Beachten Sie beim Erstellen einer Quick-Bibliothek mit QuickBASIC, ob es sich um eine neue oder um die Aktualisierung einer bereits bestehenden Bibliothek handelt. Im Fall einer Aktualisierung starten Sie QuickBASIC mit Hilfe der Option /L und geben Sie den Namen der zu aktualisierenden Bibliothek als Argument der Befehlszeile (z.B. `QB /L MeiBib`) an. Sie können auch den Namen eines Programms hinzufügen, dessen Module in die Bibliothek eingefügt werden sollen. In diesem Fall lädt QuickBASIC alle in der MAK-Datei des jeweiligen Programms angegebenen Module. Beginnen Sie die Erstellung einer neuen Bibliothek durch Aufruf von QuickBASIC, und laden Sie die in die Bibliothek zu übertragenden Dateien mit Hilfe des Befehls **Datei laden**.

Wenn Sie Ihr Programm beim Aufruf von QuickBASIC laden, entfernen Sie alle in der Quick-Bibliothek unerwünschten Module, einschließlich des Hauptmoduls (falls es keine in der Bibliothek einzusetzenden Prozeduren enthält).

16.6.1 Entfernen und Laden von Modulen

Um eine Quick-Bibliothek so kompakt wie möglich zu gestalten, sind alle unwichtigen, aktuell geladenen Module mit Hilfe nachstehender Arbeitsschritte zu entfernen:

1. Wählen Sie den Befehl **Datei entfernen** aus dem Menü **Datei**.
2. Wählen Sie das zu entfernende Modul aus dem Verzeichnisfeld aus, und drücken Sie die EINGABETASTE. Das Modul wird zwar aus dem Arbeitsspeicher gelöscht, bleibt aber als Datei bestehen.
3. Wiederholen Sie Arbeitsschritte 1 und 2 bis alle unerwünschten Module entfernt sind.

Es gibt auch die Möglichkeit, QuickBASIC mit oder ohne Angabe einer Bibliothek zu starten und nur die gewünschten Module zu laden. In diesem Fall wird jedes Modul anhand des Befehls **Datei laden** aus dem Menü **Datei** geladen.

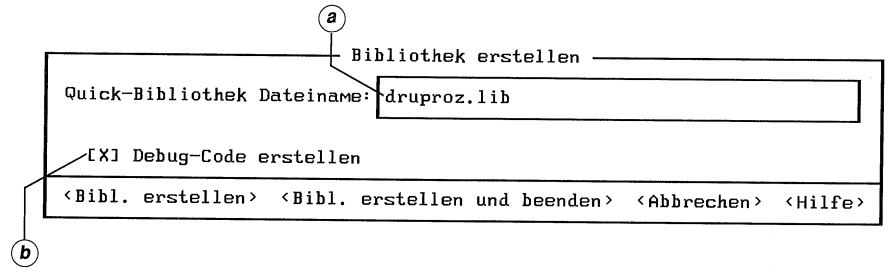
Die Module werden wie folgt einzeln geladen:

1. Wählen Sie den Befehl **Datei laden** aus dem Menü **Datei** aus.
2. Wählen Sie den Namen des gewünschten Moduls im Verzeichnisfeld aus.
3. Wiederholen Sie Arbeitsschritte 1 und 2 bis alle gewünschten Module geladen sind.

16.6.2 Die Erstellung von Bibliotheken

Nachdem Sie, falls erforderlich, die vorhergehende Bibliothek und die neuen in die Quick-Bibliothek einzufügenden Module geladen haben, wählen Sie den Befehl **Bibliothek erstellen** aus dem Menü **Ausführen**. Anschließend erscheint das in Abbildung 16.3 dargestellte Dialogfeld.

Abbildung 16.3 Das Dialogfeld Bibliothek erstellen



a) Name der Bibliothek.

b) Erteilt auf Anforderung zusätzliche Informationen zum Debuggen.

Erstellen Sie die Quick-Bibliothek nach Auswahl des Befehls **Bibliothek erstellen** wie folgt:

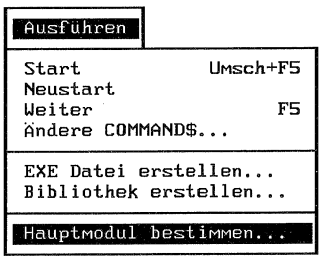
1. Geben Sie den Namen der zu erstellenden Bibliothek im Textfeld **Quick-Bibliothek Dateiname** ein.

   Wenn nur ein Basisname eingegeben wird, ergänzt ihn QuickBASIC beim Erstellen der Quick-Bibliothek mit der Erweiterung *.qlb*. Falls Sie keine Erweiterung für den Namen der Bibliothek wünschen, schließen Sie den Basisnamen mit einem Punkt (•) ab. Es gibt keine weiteren Einschränkungen für eindeutige Basisnamen und Erweiterungen, solange sie den DOS-Regeln für Dateinamen entsprechen.

2. Wählen Sie das Kontrollfeld **Debug-Code erstellen** nur dann aus, wenn Sie einen Fehler verfolgen wollen, der sich vermutlich in der zu aktualisierenden Bibliothek befindet. Diese Option vergrößert die Bibliothek, verlangsamt den Programmablauf und liefert nur wenig Informationen über Fehler.
3. Erstellen Sie die Quick-Bibliothek anhand einer der folgenden Methoden:
   - Wählen Sie die Befehlsfläche **Bibliothek erstellen** aus, wenn Sie nach Anlegen der Quick-Bibliothek in der Programmierumgebung bleiben wollen.
   - Wählen Sie die Befehlsfläche **Bibliothek erstellen und beenden** aus, wenn Sie nach Anlegen der Quick-Bibliothek QuickBASIC beenden möchten.

Hinweis Eine Quick-Bibliothek, die Ereignisse wie Tastenbetätigungen verfolgen soll, muß mindestens eine ereignisverfolgende Anweisung in einem der Module enthalten. Zu diesem Zweck genügt auch eine einfache Anweisung wie **TIMER OFF**, ohne die die Ereignisse in der Quick-Bibliothek jedoch nicht korrekt verfolgt werden können.

16.7 Der Befehl Hauptmodul bestimmen (nur im vollständigen Menü)



Dieser Befehl ermöglicht die Auswahl eines Hauptmoduls aus dem Verzeichnis der aktuell geladenen Module.

Ein einmoduliges Programm besteht *nur* aus dem Hauptmodul. Demnach wird ein Hauptmodul bereits bei Eingabe und Speichern der ersten Anweisung erstellt. In einem mehrmoduligen Programm enthält das Hauptmodul die erste während des Programmablaufs auszuführende Anweisung. Es steht in allen Verzeichnisfeldern, die Module anzeigen (wie zum Beispiel dem Dialogfeld des Befehls **SUBs**) an erster Stelle.

Wenn QuickBASIC ohne Angabe eines Dateinamens gestartet wird, zeigt die Titelleiste des Arbeitsfensters das Wort `Unbenannt` an. Der Name, unter dem der Inhalt des Arbeitsfensters (auch wenn es leer ist) gespeichert wird, wird der Name des Hauptmoduls des Programms.

Anschließend erscheint dieser Name in der Titelleiste des Arbeitsfensters.

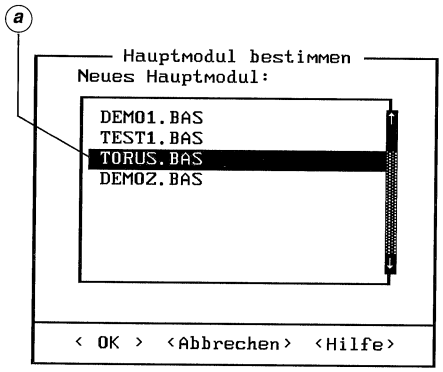
16.7.1 Ändern des Hauptmoduls

Ein Modul kann zwar in mehreren unterschiedlichen Programmen eingesetzt werden, es kann jedoch nur das Hauptmodul eines einzigen Programms bilden, welches den Namen des Moduls trägt.

Während der Bearbeitung Ihres Programms werden Sie eventuell feststellen, daß eine Umorganisierung angebracht ist und daß die Ausführung nicht mit dem aktuell festgelegten Hauptmodul beginnen soll. Zu diesem Zweck verfahren Sie wie folgt:

1. Wählen Sie den Befehl **Hauptmodul bestimmen** aus dem Menü **Ausführen**. Das Dialogfeld **Hauptmodul bestimmen** enthält ein Verzeichnis der aktuell geladenen Module.
2. Wählen Sie das Modul aus, das als Hauptmodul festgelegt werden soll, und drücken Sie die EINGABETASTE (siehe Abbildung 16.4).

Abbildung 16.4 Das Dialogfeld Hauptmodul bestimmen



a) Wird neues Hauptmodul.

16.7.2 Die MAK-Datei

Beim Speichern eines mehrmoduligen Programms erstellt QuickBASIC eine spezielle Datei, die die Namen aller Module des Programms enthält. Diese Datei nimmt den Namen des Hauptmoduls an und ist mit der Erweiterung *.mak* versehen.

Wenn das Programm neu geladen wird, findet QuickBASIC alle Module mit Hilfe dieser MAK-Datei, die aus diesem Grund nie gelöscht werden sollte. Wenn ein mehrmoduliges Programm in ein anderes Verzeichnis kopiert wird, muß auch die MAK-Datei mitkopiert werden.

Die MAK-Datei umfaßt nur die Module eines Programms, wobei das jeweilige Hauptmodul an erster Stelle steht. Die MAK-Liste kann keine anderen Dateitypen enthalten. Wenn Ihr Programm Include-Dateien benutzt, müssen die jeweiligen Dateien als Argumente eines Metabefehls **$INCLUDE** angegeben werden.

Die MAK-Datei ist eine Textdatei und wird dementsprechend mit Hilfe der zum Befehl **Datei laden** gehörenden Option **Dokument** aus dem Menü **Datei** in QuickBASIC geladen.

Die in einer MAK-Datei aufgeführten Module lassen sich entfernen, ohne die MAK-Datei selbst zu bearbeiten. Laden Sie das Programm und befolgen Sie anschließend die in Abschnitt 16.6.1, "Entfernen und Laden von Modulen", beschriebenen Arbeitsschritte. Beim Speichern des Programms werden die Namen sämtlicher entfernter Module aus der MAK-Datei gelöscht.

17 Grundkonzepte und Techniken der Fehlerbeseitigung

Dieses Kapitel führt in das Thema der Fehlerbeseitigung in QuickBASIC ein. QuickBASIC kann bei der Verfolgung von Fehlern, die sich aus Unstimmigkeiten in der Programmlogik ergeben, behilflich sein.

Dieses Kapitel befaßt sich mit folgenden Debug-Themen:

- Vermeiden von Fehlern beim Schreiben des Programms mit Hilfe von QuickBASIC.
- Begriffe und Grundkonzepte der Fehlerbeseitigung.

Weitere Informationen zum Debuggen entnehmen Sie bitte Kapitel 18, "Das Menü Debug".

17.1 Debuggen mit QuickBASIC

QuickBASIC-Einrichtungen, wie Haltepunkte, Stoppbedingungen, Aktuellen Wert anzeigen und Anzeigevariablen vereinfachen die Fehlerbeseitigung. Zu den grundlegenden Debug-Werkzeugen gehören auch die Anweisungen **STOP**, **PRINT**, **CONT**, **TRON** und **TROFF**.

In QuickBASIC sind keine separaten Hilfsprogramme oder Aufrufe von Sonderoptionen zum Debuggen des Programms erforderlich. Nach Entdecken eines Fehlers kann dieser sofort von der Umgebung aus verfolgt werden. QuickBASIC bietet alle dazu nötigen Hilfsmittel.

Darüber hinaus ist es in der QuickBASIC-Umgebung nicht notwendig, ein Programm nach einer Änderung erneut zu kompilieren oder zu binden. Normalerweise können Sie ein Programm bis zur Problemstelle ausführen, die Ausführung unterbrechen, den Fehler beseitigen und den Programmablauf problemlos fortsetzen. In QuickBASIC verlaufen Programmieren und Debuggen parallel.

17.2 Fehlervermeidung anhand von QuickBASIC

Vermeidung von Fehlern ist die beste Methode ein fehlerfreies Programm zu erhalten. Auf folgende drei Arten lassen sich Fehler am besten vermeiden:

1. Sorgfältige Entwicklung des Programms.

   Eine wichtige Faustregel für die Entwicklung eines guten Programms ist die Aufteilung verschiedener Aufgaben auf **SUB**- oder **FUNCTION**-Prozeduren, die jede Aufgabe einzeln übernehmen. Es ist wesentlich einfacher, eine kleine Prozedur zu debuggen, als ein umfangreiches Programm ohne Prozeduren.

   Ein weiterer Vorteil der Prozeduren besteht in der Möglichkeit, Prozedurengruppen in einer getrennten, als Modul bezeichneten Datei abzuspeichern und sie in anderen Programmen zur Durchführung derselben Aufgaben einzusetzen. (Weitere Informationen zu Prozeduren und Modulen sind Kapitel 2, "Prozeduren: Unterprogramme und Funktionen", in *Programmieren in BASIC* zu entnehmen.)

2. Verwendung des Direktfensters.

   Verwenden Sie während der Programmierung das am unteren Ende des QuickBASIC-Bildschirms vorhandene Direktfenster, um Bestandteile des Programms zu isolieren und zu testen. Näheres zum Direktfenster erfahren Sie im Abschnitt 17.3.5. Wenn die einzelnen Bestandteile funktionieren, können sie im Arbeitsfenster eingefügt werden.

3. Häufige Programmausführung.

   Da QuickBASIC nach Eingabe jeder Zeile und Betätigung der EINGABETASTE jede einzelne Anweisung überprüft und in ausführbaren Code (für den Computer verständliche Anweisungen) umwandelt, können Sie das Programm beim Hinzufügen neuer Teile fortlaufend testen. Dieser Vorgang ist bei der Beseitigung kleiner Fehler behilflich, die in einem abgeschlossenen Programm schwer zu verfolgen wären.

Wenn das Programm beispielsweise folgenden Code enthält:

```
DIM Datenfeld$ (1 TO 20)
.
.
.
I% = 1
DO
   INPUT Temp$
   IF Temp$ <> "" THEN
      Datenfeld$(I%) = Temp$
      I% = I + 1
   END IF
LOOP UNTIL Temp$ = "" OR I% > 20
```

der eine logische Schleife darstellen soll, können Sie diesen Codeabschnitt wie folgt testen:

1. Führen Sie die Schleife aus, und geben Sie den Text nach der Eingabeaufforderung ein. (Drücken Sie die EINGABETASTE, anstatt Text einzugeben, wenn Sie die Schleife abschließen möchten.)
2. Geben Sie folgende Zeile im Direktfenster ein:

```
For I% = 1 to 20: PRINT Datenfeld$(I%): NEXT
```

Die Ausgabe wird Sie auf eine Unstimmigkeit im Programm aufmerksam machen, nämlich, daß die Variable `I%` nicht inkrementiert. Bei näherer Überprüfung werden Sie merken, daß die Zeile

```
I% = I + 1
```

in

```
I% = I% + 1
```

umgeändert werden muß.

Außer der sorgfältigen Entwicklung des Programms, der Verwendung des Direktfensters und der häufigen Programmausführung gehört auch das Anzeigen von Variablen oder Ausdrücken zu den empfehlenswerten Debug-Techniken, die das "Überwachen" des Wertes der Variablen `I%` während der Ausführung der Schleife erlaubt. Näheres zum Überwachen eines Ausdruckes während des Programmablaufes finden Sie in Abschnitt 17.3.3, "Das Anzeigen von Variablen und Ausdrücken", und Abschnitt 17.3.4, "Das Debug-Fenster".

17.3 Die QuickBASIC-Einrichtungen zum Debuggen

Dieser Abschnitt definiert einige grundlegende Begriffe und Konzepte der Fehlerbeseitigung. Einzelheiten zum Menü **Debug** und zur Anwendung der Debug-Befehle von QuickBASIC sind Kapitel 18, "Das Menü Debug", zu entnehmen.

17.3.1 Die Ablaufverfolgung (nur im vollständigen Menü)

Der Befehl **Verfolgen ein** hebt jede Anweisung während der Ausführung hervor. Wenn das Programm den Bildschirm manipuliert, wechselt QuickBASIC zwischen den hervorgehobenen Anweisungen und dem Ausgabebildschirm (dieser Vorgang wird als "animierte Ausführung" bezeichnet). Mit Hilfe der Ablaufverfolgung kann beispielsweise die Verzweigung in einer **IF...THEN...ELSE**-Anweisung verfolgt werden. Die QuickBASIC-Optionen zur Ablaufverfolgung ermöglichen die Ausführung des gesamten Programms (normale Ablaufverfolgung) sowie die Ausführung jeder Anweisung oder Prozedur im Einzelschrittmodus. Der Einzelschrittmodus wird in Abschnitt 17.3.6, "Andere Debug-Einrichtungen", eingehend erläutert.

Das Einschalten der Ablaufverfolgung aktiviert automatisch den Befehl **Rückverfolgen ein**, der die letzten 20 ausgeführten Anweisungen erfaßt und Ihnen die Möglichkeit bietet, diese Anweisungen mit Hilfe der Befehle **Verfolgen vorwärts** und **Verfolgen rückwärts** nach vorne und nach rückwärts durchzugehen.

Siehe auch

Abschnitt 18.6, "Der Befehl Verfolgen ein", und Abschnitt 18.7, "Der Befehl Rückverfolgen ein".

17.3.2 Haltepunkte und Stoppbedingungen

Ein Haltepunkt bezeichnet eine Stelle im Programm, an der die Ausführung angehalten werden soll. Haltepunkte ermöglichen eine beliebige Teilausführung des Programms, durch die der betreffende Programmabschnitt getestet werden kann. Sie können das Programm auch an entscheidenden Punkten anhalten, um den Wert einer Variablen zu untersuchen.

Eine Stoppbedingung ist ein Ausdruck, der das Programm anhält, sobald der Ausdruck wahr wird (ungleich Null). Es ist zum Beispiel empfehlenswert, eine Stoppbedingung einzusetzen, wenn das Programm bei einer Schleife abstürzt, die `X` von `1` bis `50` inkrementiert. Die Stoppbedingung `X=49` hält das Programm an, wenn `X` den Wert `49` annimmt. Sie können dann die 49. Schleife im Einzelschrittmodus durchgehen, um die Anweisung, die den Programmabsturz verursacht hat, zu lokalisieren.

Anzeigevariablen erscheinen im Debug-Fenster, das alle aktuellen Stoppbedingungen und Anzeigevariablen anzeigt.

Siehe auch

Abschnitt 17.3.4, "Das Debug-Fenster", Abschnitt 18.1, "Der Befehl Variable anzeigen", und Abschnitt 18.8, "Der Befehl Haltepunkt ein/aus".

17.3.3 Das Anzeigen von Variablen und Ausdrücken

Die mit dem Befehl **Variable anzeigen** eingegebenen Anzeigevariablen erlauben die Überwachung eines Variablenwertes (in Form einer Zahl oder einer Zeichenkette) oder der Bedingung eines Ausdruckes (wahr oder falsch) während der Programmausführung. Wenn Sie zum Beispiel den Wert der Variablen `MeinVar` und des Ausdruckes `X=1` verfolgen möchten, setzt QuickBASIC die Variable und den Ausdruck im Debug-Fenster ein. Während des Programmablaufs zeigt QuickBASIC die verschiedenen Werte von `MeinVar` und die Bedingung von `X=1` (-1 für wahr und 0 für falsch) im Debug-Fenster an. Weist das Programm *MeinProg.bas* `MeinVar` den Wert `10` und `X` den Wert `1` zu, erscheint im Debug-Fenster

```
MEINPROG.BAS  MeinVar:  10
MEINPROG.BAS  X=1:      -1
```

Wenn das Programm den Wert von `MeinVar` auf `99` und den Wert von `X` auf `3` ändert, zeigt das Debug-Fenster folgendes an:

```
MEINPROG.BAS  MeinVar:  99
MEINPROG.BAS  X=1:       0
```

Der Befehl **Aktuellen Wert anzeigen** gibt den Wert einer Variablen oder eines Ausdruckes, bei der/dem der Cursor steht, in einem Dialogfeld aus, ohne zuvor das Debug-Fenster öffnen zu müssen. Die Option **Aktuellen Wert anzeigen** bietet Ihnen jedoch auch die Möglichkeit, die Variable oder den Ausdruck im Debug-Fenster einzusetzen.

Siehe auch

Abschnitt 17.3.4, "Das Debug-Fenster", Abschnitt 18.1, "Der Befehl Variable anzeigen", und Abschnitt 18.2, "Der Befehl Aktuellen Wert anzeigen".

17.3.4 Das Debug-Fenster

Das am oberen Ende des QuickBASIC-Bildschirms geöffnete Debug-Fenster gewährleistet die Verfolgung der Stoppbedingungen und die Überwachung der Ausdrücke während des Programmablaufs.

Der aktuelle Wert oder die aktuelle Bedingung (wahr oder falsch) einer Variablen oder eines Ausdruckes erscheinen im Debug-Fenster nur während der Ausführung des Programmteiles, in dem die Stoppbedingung oder die Anzeigevariable hinzugefügt wurde. Während des restlichen Programmablaufs erscheint die Meldung `Nicht anzeigbar`.

Nehmen wir zum Beispiel an, daß Sie während der Arbeit an einer Prozedur mit dem Namen `MeinSub` eine Anzeigevariable erstellen, um den Wert der Variablen `MeinVar` zu überwachen. Der Wert dieser Variablen wird nur während der Ausführung der Prozedur `MeinSub` im Debug-Fenster angezeigt. Während der Ausführung des Modul-Ebenen-Codes oder anderer Prozeduren steht im Debug-Fenster die Meldung `Nicht anzeigbar` nach `MeinVar`.

Auch wenn in einem Programm eine Variable mit Hilfe der Anweisung **SHARED** auf Modul- und Prozedurebene gemeinsam benutzt wird, ist die Variable nur auf der Ebene verfolgbar, die beim Erstellen der Anzeigevariablen im Debug-Fenster aktiv war.

Wenn eine **DEF FN**-Funktion aktiv ist, können keine Programmelemente verfolgt werden.

Siehe auch

Abschnitt 17.3.3, "Das Anzeigen von Variablen und Ausdrücken", Abschnitt 18.1, "Der Befehl Variable anzeigen", und Abschnitt 18.2, "Der Befehl Aktuellen Wert anzeigen".

17.3.5 Das Direktfenster

Das Direktfenster erscheint am unteren Ende des Bildschirms. In diesem Fenster können BASIC-Anweisungen direkt ausgeführt werden, um für die Fehlerbeseitigung verwertbare Informationen zu erhalten.

Es ist jedoch zu beachten, daß das Direktfenster nur auf Variablen, die sich im momentan aktiven Programmteil (Modul oder Prozedur) befinden, zugreifen kann.

Wenn beispielsweise ein Programm in einer Prozedur unterbrochen wird, hat das Direktfenster auf die von der **SHARED**-Anweisung angegebenen, lokalen und globalen Variablen und Ausdrücke Zugriff, nicht jedoch auf Variablen und Ausdrücke, die zum Modul-Ebenen-Code oder zu einer anderen Prozedur lokal sind.

Siehe auch

Abschnitt 10.5, "Verwendung des Direktfensters".

17.3.6 Andere Debug-Einrichtungen

Tabelle 17.1 beschreibt zusätzliche in der Version 4.5 von QuickBASIC verfügbare Einrichtungen zum Debuggen. Einige können nur zusammen verwendet werden, während andere üblicherweise selbständig eingesetzt werden.

Tabelle 17.1 Zusätzliche Einrichtungen zum Debuggen

Einrichtung	*Ausführung*	*Beschreibung*
Der Befehl **Halt bei Fehler**	Auswahl aus dem Menü **Debug**	Entdeckt Fehler im Programm anhand einer Fehlerbehandlungsroutine. Beim Auftreten eines Fehlers deaktiviert dieser Befehl die Fehlerbehandlung und hält die Programmausführung an. Drücken Sie die Tastenkombination UMSCHALT+F8 (**Rückverfolgen ein**), um die fehlerverursachende Anweisung hervorzuheben.
Bis zur Cursorposition ausführen	F7 betätigen	Führt das Programm von Anfang bis zu einschließlich der Zeile aus, die den Cursor enthält, und unterbricht den Ablauf an dieser Stelle. Diese Einrichtung dient der schnellen Programmausführung bis zu einem problematischen Abschnitt, um dann den Fehler schrittweise verfolgen zu können.
Der Befehl **Nächste Anweisung festlegen**	Auswahl aus dem Menü **Debug**	Ändert die Ausführungsfolge eines unterbrochenen Programms, um die nächste auszuführende Anweisung an der Cursorstelle festzulegen. Enthält das Programm einen Fehler, der sich sofort beseitigen läßt, können Sie QuickBASIC anhand dieses Befehls anweisen, die Ausführung bei Auswahl des Befehls **Weiter** ab der korrigierten Zeile fortzusetzen.
Einzelschrittmodus	F8 betätigen	Führt eine einzige Anweisung aus und ist beim Debuggen von Code besonders nützlich. Die kombinierte Anwendung des Einzelschrittmodus und des Debug-Fensters bietet eine hervorragende Möglichkeit zur Untersuchung der Auswirkung jeder Anweisung auf einzelne Variablen.

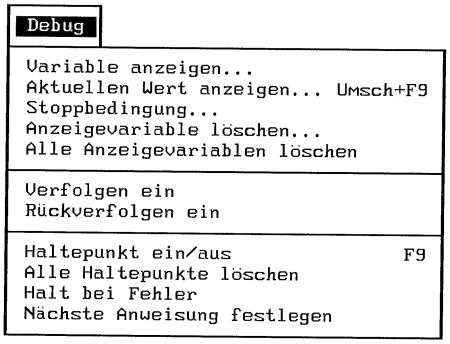
Fortsetzung auf der folgenden Seite.

Einrichtung	*Ausführung*	*Beschreibung*
Prozedurschrittmodus	F10 betätigen	Führt einen Prozeduraufruf und die betreffende Prozedur als eine einzige Anweisung aus. Wenn Sie bei Anwendung des Einzelschritt-modus eine Prozedur antreffen, die bereits funktioniert, erlaubt Ihnen der Prozedurschrittmodus, die gesamte Prozedur auf einmal (anstatt schrittweise) auszuführen.

Siehe auch

Abschnitt 18.10, "Der Befehl Halt bei Fehler", und Abschnitt 18.11, "Der Befehl Nächste Anweisung festlegen".

18 Das Menü Debug



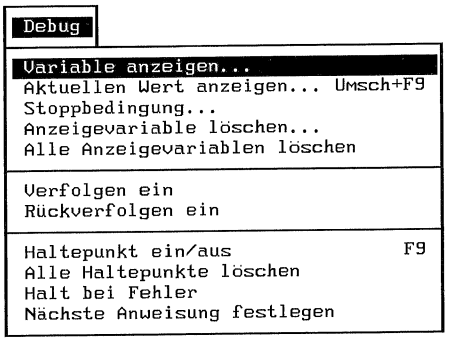
Die Einrichtungen im Menü **Debug** beschleunigen die Fehlerbeseitigung im Programm und ermöglichen die Überwachung der veränderlichen Werte von Variablen und Ausdrücken, die Verfolgung der Programmausführung, die Unterbrechung des Programms an einer bestimmten Stelle oder bei Erfüllung gewisser Bedingungen sowie das Festlegen der Anweisung, mit der ein unterbrochenes Programm fortgesetzt wird.

Das Menü **Debug** umfaßt folgende Befehle:

- **Variable anzeigen**. Zeigt während des Programmablaufes die Werte von Variablen und Ausdrücken im Debug-Fenster an.
- **Aktuellen Wert anzeigen**. Liefert den aktuellen Wert einer Variablen oder eines Ausdruckes.
- **Stoppbedingung**. Hält das Programm an, wenn eine gegebene Bedingung wahr wird (nur im vollständigen Menü).
- **Anzeigevariable löschen**. Entfernt ein Element aus dem Debug-Fenster.
- **Alle Anzeigevariablen löschen**. Löscht alle Einträge aus dem Debug-Fenster und schließt es (nur im vollständigen Menü).
- **Verfolgen ein**. Geht das Programm schrittweise durch, indem es die aktuell ausgeführte Anweisung hervorhebt (nur im vollständigen Menü).

- **Rückverfolgen ein**. Erfaßt die letzten 20 vom Programm ausgeführten Anweisungen (nur im vollständigen Menü).
- **Haltepunkt ein/aus**. Schaltet Haltepunkte ein und aus.
- **Alle Haltepunkte löschen**. Löscht alle zuvor festgelegten Haltepunkte.
- **Halt bei Fehler**. Setzt einen impliziten Haltepunkt bei der zuletzt angetroffenen Fehlerbehandlungsroutine (nur im vollständigen Menü).
- **Nächste Anweisung festlegen**. Ändert die Ausführungsfolge des Programms, so daß die an der Cursorstelle stehende Anweisung als Nächste ausgeführt wird (nur im vollständigen Menü).

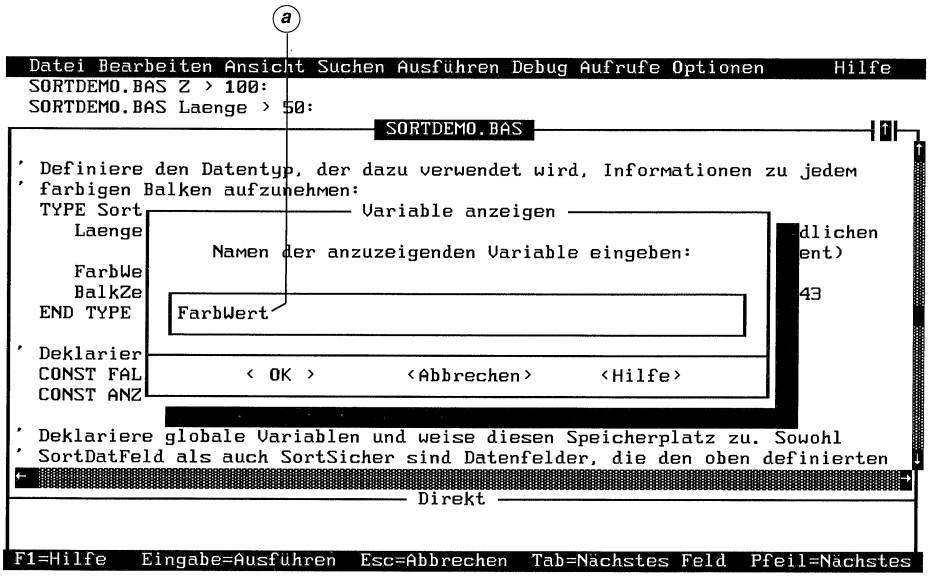
18.1 Der Befehl Variable anzeigen



Dieser Befehl zeigt die Werte von Variablen und Ausdrücken während des Programmablaufes an. QuickBASIC öffnet dazu ein Debug-Fenster am oberen Ende des Bildschirms.

Wählen Sie **Variable anzeigen** aus dem Menü **Debug** aus, um den Befehl zu aktivieren und ein Dialogfeld erscheinen zu lassen, in dem der Name der zu verfolgenden Variablen oder des Ausdruckes einzugeben ist (siehe Abbildung 18.1).

Abbildung 18.1 Das Dialogfeld Variable anzeigen



a) Variable oder Ausdruck.

Das Überwachen einer Variablen im Debug-Fenster vermeidet die Verfolgung des Variablenwertes durch wiederholte Anwendung von **PRINT**-Anweisungen. Im Debug-Fenster kann ebenfalls die Bedingung (wahr oder falsch) eines Ausdruckes verfolgt werden, wobei wahr durch `-1` und falsch durch `0` dargestellt wird. Nach Abschluß der Fehlerbeseitigung lassen sich die verfolgten Variablen oder Ausdrücke aus dem Debug-Fenster entfernen.

Während der Programmausführung entsprechen die Änderungen des im Debug-Fenster angezeigten Wertes den Wertänderungen im Programm. Es kann vorkommen, daß die Meldung `Nicht anzeigbar` anstelle eines Wertes erscheint. Das bedeutet, daß der Debugger auf die betreffende Variable keinen Zugriff hat, da zu diesem Zeitpunkt nicht der Programmteil (der Prozedur- oder Modul-Ebenen-Code) ausgeführt wird, aus dem die Variable im Debug-Fenster eingesetzt wurde. Demnach ist eine Variable nur während der Ausführung des Programmteils anzeigbar, aus dem sie im Debug-Fenster eingefügt wurde.

Sollen jedoch nur einzelne Werte einer Variablen angezeigt werden, ist es einfacher, den in Abschnitt 18.2 beschriebenen Befehl **Aktuellen Wert anzeigen** zu benutzen.

Beispiel

Im nachstehenden Beispiel wird die *n*te Zahl einer als Fibonacci-Folge bekannten Zahlenfolge berechnet. In dieser mit 1 und 2 beginnenden Zahlenfolge stellt jede Zahl die Summe der zwei vorhergehenden Zahlen dar. Die ersten 10 Zahlen der Fibonacci-Folge sind 1, 2, 3, 5, 8, 13, 21, 34, 55 und 89. Das Beispiel veranschaulicht die Anwendung des Befehls **Variable anzeigen** und die animierte Ablaufverfolgung (das abwechselnde Umschalten zwischen hervorgehobenen Anweisungen im Programm und dem Ausgabebildschirm).

1. Starten Sie QuickBASIC oder wählen Sie den Befehl **Neues Programm** aus dem Menü **Datei** aus, und geben Sie folgendes ein:

```
DEFINT A-Z
INPUT n
PRINT Fib(n)
END
```

2. Wählen Sie den zum Menü **Debug** gehörenden Befehl **Variable anzeigen** aus und geben Sie den Buchstaben *n* im Textfeld ein, um den auf Modulebene angenommen Wert von *n* anzuzeigen.

3. Erstellen Sie eine **FUNCTION**-Prozedur mit dem Namen `Fib`, indem Sie den Befehl **Neue FUNCTION** aus dem Menü **Bearbeiten** auswählen und die Bezeichnung `Fib` im Dialogfeld eingeben. (Aufgrund der im 1. Arbeitsschritt eingegebenen **DEF***typ*-Anweisung wird die Zeile `DEFINT A-Z` automatisch oberhalb der ersten Zeile der **FUNCTION**-Definition eingefügt.) Geben Sie anschließend die Prozedur wie folgt ein:

```
FUNCTION Fib(n)
    IF n > 2 THEN
        Fib = Fib(n - 1) + Fib(n - 2)
    ELSE Fib = n
    END IF
END FUNCTION
```

4. Wählen Sie den Befehl **Variable anzeigen** aus dem Menü **Debug** aus, und geben Sie erneut `n` im Debug-Fenster ein, um den innerhalb der Prozedur angenommenen Wert von *n* anzuzeigen.

5. Wählen Sie den zum Menü **Debug** gehörenden Befehl **Verfolgen ein**.

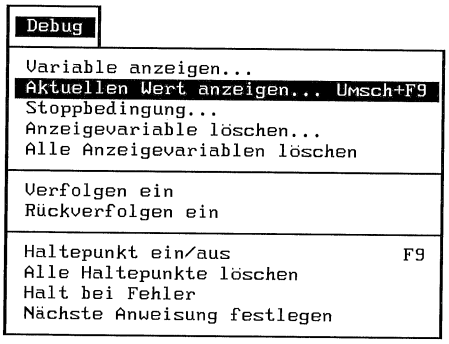
6. Führen Sie das Programm mit Hilfe des zum Menü **Ausführen** gehörenden Befehls **Start** aus.

7. Geben Sie die Zahl 8 nach der Eingabeaufforderung ein, und beobachten Sie dabei das Debug-Fenster, das die Änderungen von `n` anzeigt, während die Prozedur die achte Zahl der Fibonacci-Folge durch Rekursion berechnet.
8. Sehen Sie sich die Ergebnisse des Programms mit Hilfe des Befehls **Ausgabebildschirm** (oder durch Betätigen der Taste F4) auf dem Bildschirm an.

Siehe auch

Abschnitt 17.3.3, "Das Anzeigen von Variablen und Ausdrücken", und Abschnitt 17.3.4, "Das Debug-Fenster".

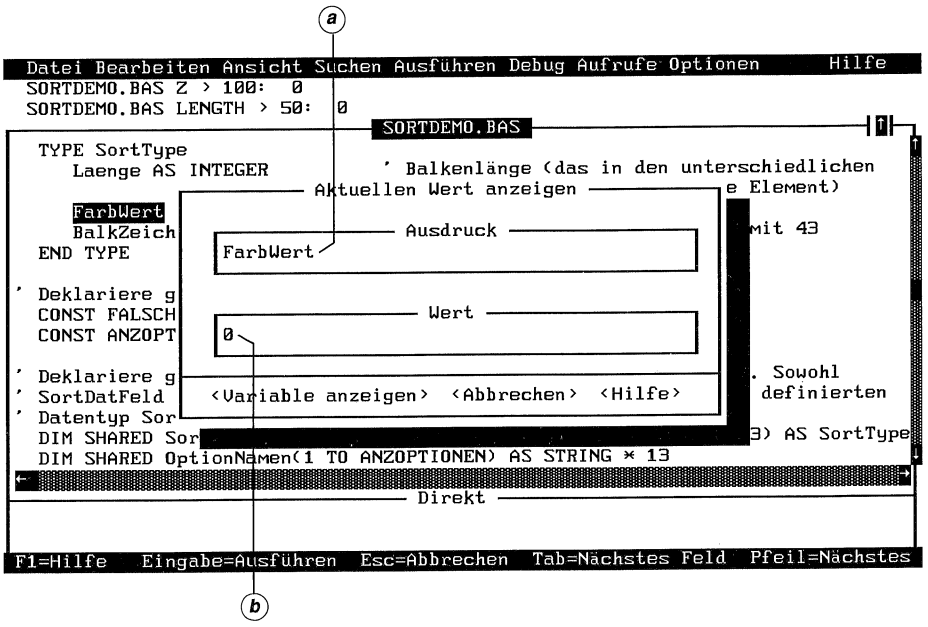
18.2 Der Befehl Aktuellen Wert anzeigen



Dieser Befehl zeigt den Wert einer Variablen oder die Bedingung eines Ausdruckes (wahr oder falsch) in einem Dialogfeld an.

Benutzen Sie diesen Befehl, indem Sie den Cursor innerhalb (oder rechts von) der Variablen plazieren, oder wählen Sie den gewünschten Ausdruck und den Befehl **Aktuellen Wert anzeigen** aus dem Menü **Debug** aus. Anschließend zeigt QuickBASIC ein Dialogfeld an, das im oberen Textfeld die Variable oder den Ausdruck, und im unteren Textfeld den Wert bzw. die Bedingung (-1 für wahr und 0 für falsch) enthält (siehe Abbildung 18.2).

Abbildung 18.2 Das Dialogfeld Aktuellen Wert anzeigen



a) Variable oder Ausdruck.

b) Wert oder Bedingung.

Der Befehl **Aktuellen Wert anzeigen** wird zum Debuggen von Zuweisungsanweisungen und zum Überprüfen von Variablen oder Ausdrücken verwendet. Soll dagegen eine Variable oder ein Ausdruck fortlaufend überwacht werden, empfiehlt es sich, das betreffende Element mit Hilfe des Befehls **Variable anzeigen** (oder durch Betätigen der Befehlsfläche **Variable anzeigen** aus dem Dialogfeld **Aktuellen Wert anzeigen**) im Debug-Fenster einzusetzen.

Der Befehl **Aktuellen Wert anzeigen** kommt besonders beim Anzeigen eines Wertes oder einer Bedingung in einem unterbrochenen Programm zur Anwendung.

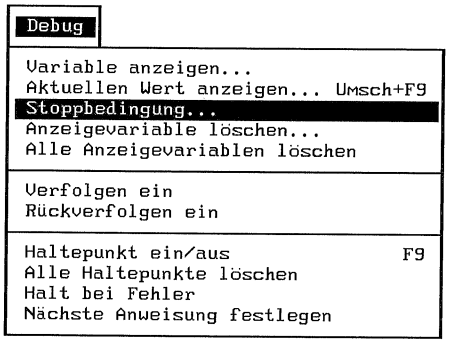
Tastenkombination zur direkten Befehlseingabe

UMSCHALT+F9

Siehe auch

Abschnitt 17.3.3, "Das Anzeigen von Variablen und Ausdrücken", und Abschnitt 17.3.4, "Das Debug-Fenster".

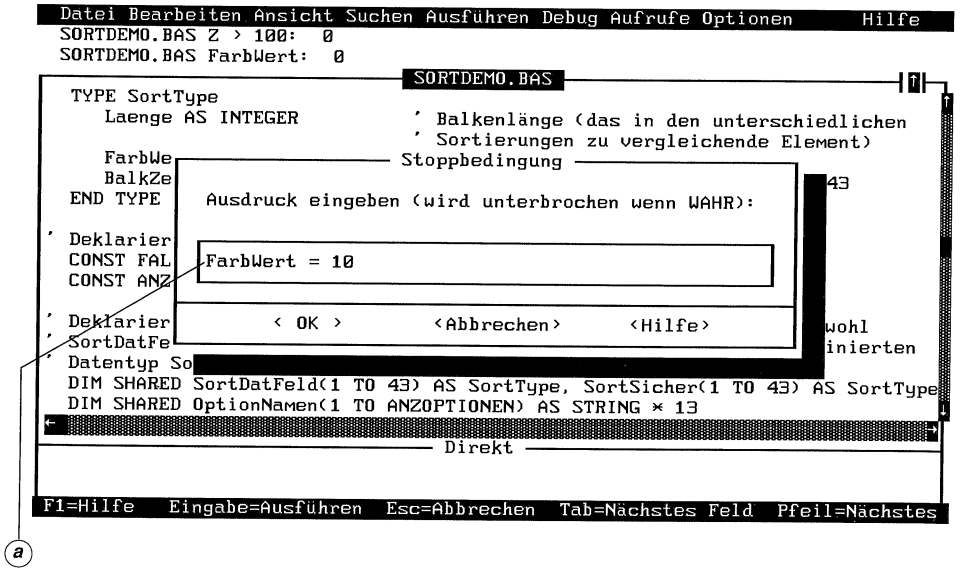
18.3 Der Befehl Stoppbedingung (nur im vollständigen Menü)



Anhand dieses Befehls kann eine Stoppbedingung festgelegt werden, mit anderen Worten ein Ausdruck, der die Programmausführung anhält, sobald der jeweilige Ausdruck wahr wird (wenn beispielsweise eine Variable X den Wert 100 erreicht). Die betreffende Stoppbedingung wird im Debug-Fenster angezeigt.

Bei Auswahl des Befehls **Stoppbedingung** erscheint ein Dialogfeld, in dessen Textfeld ein beliebiger Variablenname oder Ausdruck mit den entsprechenden Vergleichsoperatoren wie in Abbildung 18.3 eingegeben werden kann.

Abbildung 18.3 Das Dialogfeld Stoppbedingung



a) Bedingung zum Anhalten des Programms.

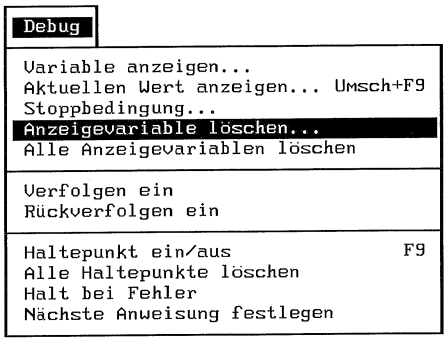
Die Vergleichsoperatoren von QuickBASIC umfassen =, <>, >, <, >= und <=. Wird ein Ausdruck ohne eine entsprechende Vergleichsanweisung eingegeben, geht QuickBASIC vom Vergleich <> 0 aus und unterbricht die Programmausführung, sobald der Ausdruck einen Wert ungleich Null annimmt.

Wird eine Stoppbedingung in einer **SUB**- oder **FUNCTION**-Prozedur gesetzt, bezeichnet das Debug-Fenster den jeweiligen Ausdruck nur dann als `WAHR` oder `FALSCH`, wenn die entsprechende Prozedur ausgeführt wird. Während der restlichen Programmausführung erscheint im Debug-Fenster die Meldung `Nicht anzeigbar` neben der Stoppbedingung. Ebenso erscheint während der Ausführung einer Prozedur die Meldung `Nicht anzeigbar` neben Stoppbedingungen der Modul-Ebene.

Siehe auch

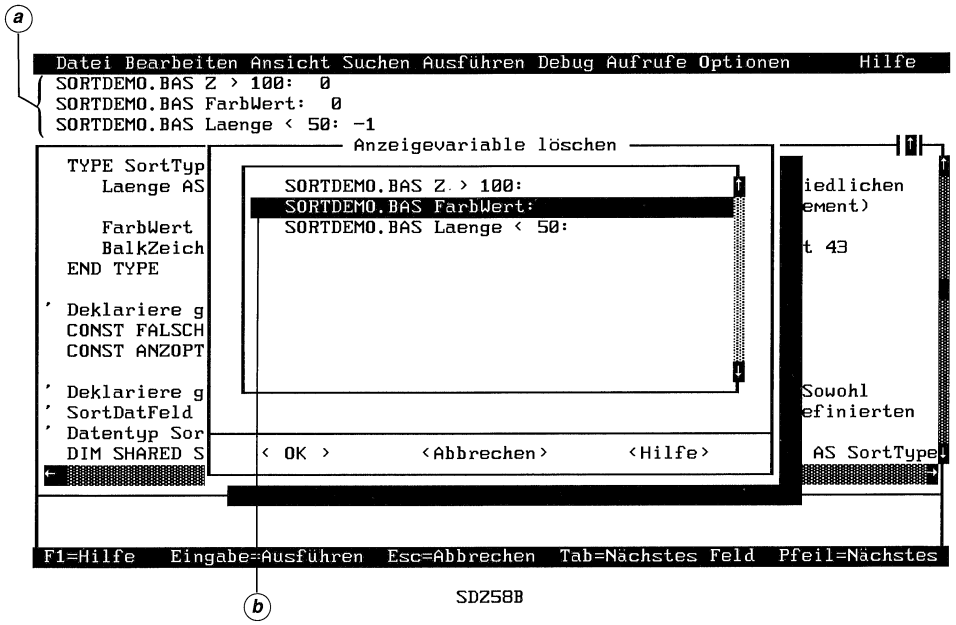
Abschnitt 17.3.2, "Haltepunkte und Stoppbedingungen", Abschnitt 18.1, "Der Befehl Variable anzeigen", und Abschnitt 18.4, "Der Befehl Anzeigevariable löschen".

18.4 Der Befehl Anzeigevariable löschen



Mit Hilfe dieses Befehls wird ein Element aus dem Debug-Fenster gelöscht. Das in Abbildung 18.4 dargestellte Dialogfeld **Anzeigevariable löschen** zeigt die aktuellen im Debug-Fenster vorhandenen Elemente an, die zum Löschen ausgewählt werden können.

Abbildung 18.4 Das Dialogfeld Anzeigevariable löschen



a) Verzeichnis der momentan im Debug-Fenster vorhandenen Elemente.

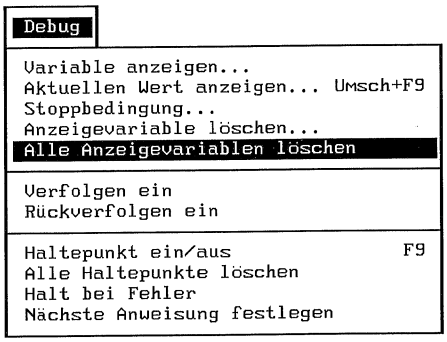
b) Zu entfernendes Element.

Löschen Sie alle Elemente des Debug-Fensters auf einmal mit Hilfe des Befehls **Alle Anzeigevariablen löschen**.

Siehe auch

Abschnitt 18.5, "Der Befehl Alle Anzeigevariablen löschen".

18.5 Der Befehl Alle Anzeigevariablen löschen (nur im vollständigen Menü)



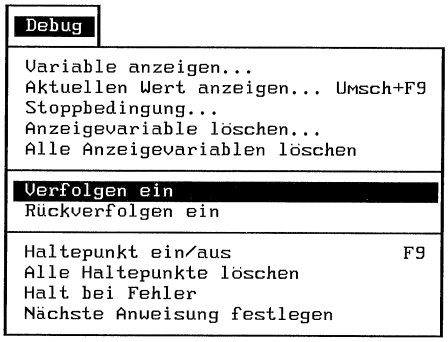
Dieser zum Menü **Debug** gehörende Befehl löscht alle Einträge aus dem Debug-Fenster und schließt dieses. Der Befehl wird üblicherweise nur dann eingesetzt, wenn der bearbeitete Programmteil richtig funktioniert.

Einzelne Elemente lassen sich jedoch anhand des Befehls **Anzeigevariable löschen** aus dem Debug-Fenster entfernen.

Siehe auch

Abschnitt 18.4, "Der Befehl Anzeigevariable löschen".

18.6 Der Befehl Verfolgen ein (nur im vollständigen Menü)



Der Befehl **Verfolgen ein** führt das Programm schrittweise aus und hebt die ausgeführten Anweisungen einzeln hervor. Ein neben der Bezeichnung dieses Befehls im Menü **Debug** stehender Punkt (•) deutet darauf hin, daß der Befehl aktiv ist. Er kann durch erneute Auswahl im Menü ausgeschaltet werden.

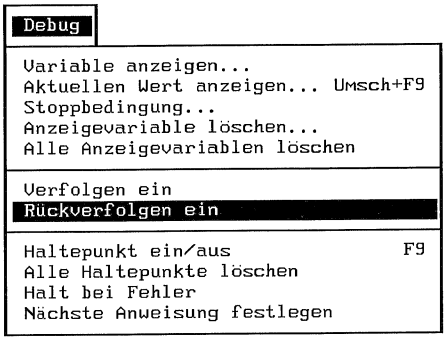
Dieser Befehl, der auch mit Stoppbedingungen, Haltepunkten und der Tastenkombination STRG+UNTBR (die die Ausführung aussetzt) kombiniert werden kann, erlaubt dem Programmierer festzustellen, ob der allgemeine Programmablauf den eigenen Vorstellungen entspricht. Darüber hinaus erfaßt er die letzten 20 ausgeführten Anweisungen, die anschließend anhand der Befehle **Verfolgen rückwärts** und **Verfolgen vorwärts** verfolgt werden können (siehe Abschnitt 18.7.1 und 18.7.2).

Ist der Befehl **Verfolgen ein** während einer Programmausgabe aktiv, wird dauernd zwischen Ausgabebildschirm und Programmierumgebung gewechselt. Mit Hilfe des Befehls **Ausgabebildschirm** aus dem Menü **Ansicht** können Sie den Bildschirm bei einem angehaltenen Programm wechseln (siehe Abschnitt 14.5).

Siehe auch

Abschnitt 18.1, "Der Befehl Variable anzeigen", Abschnitt 18.7, "Der Befehl Rückverfolgen ein", und Abschnitt 18.8, "Der Befehl Haltepunkt ein/aus".

18.7 Der Befehl Rückverfolgen ein (nur im vollständigen Menü)



Wählen Sie diesen Befehl aus dem Menü **Debug** aus, um die letzten 20, während des nächsten Programmablaufs ausgeführten Zeilen zu erfassen. Wenn dieser Befehl aktiv ist, ist er durch einen Punkt (•) neben seiner Bezeichnung im Menü **Debug** gekennzeichnet. Dieser Befehl verlangsamt die Ausführung des Programms.

Der Befehl **Rückverfolgen ein** ermöglicht die Ablaufverfolgung der letzten 20 ausgeführten Anweisungen und ist besonders dann nützlich, wenn die einem Laufzeitfehler vorangehenden Anweisungen untersucht werden sollen. Die Ablaufverfolgung ermöglicht es Ihnen auch die Verzweigungen eines Programms in verschachtelten Folgen von bedingten Strukturen, wie es bei den Anweisungen **IF...THEN...ELSE** der Fall ist, zu überprüfen. Benutzen Sie den Befehl **Rückverfolgen ein** in Verbindung mit Haltepunkten, um bis zu 20 aufeinanderfolgende Anweisungen zurückzuverfolgen.

Dieser Befehl wird, wie in den Abschnitten 18.7.1 und 18.7.2 erläutert, mit den Befehlen **Verfolgen rückwärts** und **Verfolgen vorwärts** angewandt.

Hinweis Mit Hilfe des Befehls **Rückverfolgen ein** können vorhergehende Ausführungsfolgen zwar angezeigt, jedoch nicht ausgeführt werden.

18.7.1 Verfolgen rückwärts

Dieser anhand der Tastenkombination UMSCHALT+F8 aktivierte Befehl geht die letzten 20 ausgeführten Anweisungen, die mit Hilfe des Befehls **Rückverfolgen ein** oder **Verfolgen ein** erfaßt wurden, schrittweise rückwärts durch.

Wenn Sie den Ablauf eines Programms zurückverfolgen möchten, unterbrechen Sie die Ausführung mit Hilfe der Tastenkombination STRG+UNTBR, und überprüfen Sie die letzten 20 ausgeführten Anweisungen durch Betätigen der Tastenkombination UMSCHALT+F8. Dieses Verfahren zeigt die Anweisungen des Programmablaufs der Reihe nach an und eignet sich besonders zum Verfolgen von bedingten Verzweigungen.

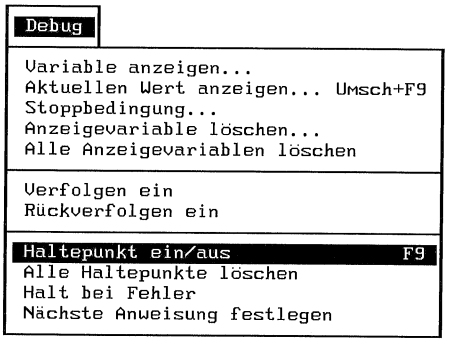
18.7.2 Verfolgen vorwärts

Dieser anhand der Tastenkombination UMSCHALT+F10 aktivierte Befehl geht die letzten 20 ausgeführten Anweisungen nach vorne durch. Er kann nur nach Ausführung des Befehls **Verfolgen rückwärts** ausgeführt werden. Da **Verfolgen rückwärts** die Ausführungsfolge umkehrt, empfiehlt es sich, im Programm zuerst schrittweise zurückzugehen, um anschließend den normalen Ablauf mit Hilfe des Befehls **Verfolgen vorwärts** zu verfolgen.

Siehe auch

Abschnitt 18.6, "Der Befehl Verfolgen ein".

18.8 Der Befehl Haltepunkt ein/aus



Dieser Befehl schaltet in der Zeile, in der sich der Cursor befindet, einen Haltepunkt (eine bestimmte Stelle im Programm, an der die Ausführung angehalten wird) ein oder aus. Zeilen, die einen Haltepunkt enthalten, werden invers dargestellt.

Es ist empfehlenswert, an Programmstellen, an denen Sie Probleme erwarten, Haltepunkte zu setzen und dann die Variablenwerte im Direktfenster zu testen, um eventuell auftretende Fehler aufzudecken. Es ist ebenfalls möglich, einen Haltepunkt vor problematischen Bereichen festzulegen und dann Variablenwerte und Programmablauf schrittweise zu überprüfen (eine Beschreibung des Einzelschrittmodus finden Sie in Abschnitt 17.3.6, "Andere Debug-Einrichtungen").

Wählen Sie den Befehl **Haltepunkt ein/aus** erneut aus, um den in der aktuellen Zeile vorhandenen Haltepunkt zu löschen.

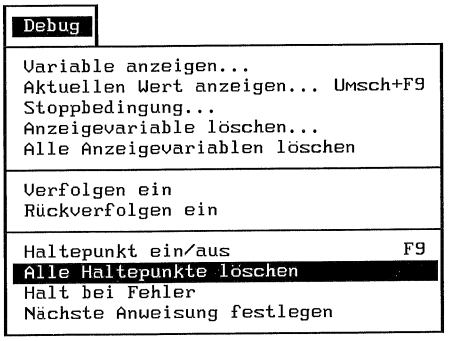
Taste zur direkten Befehlseingabe

F9

Siehe auch

Abschnitt 17.3.2, "Haltepunkte und Stoppbedingungen", und Abschnitt 18.9, "Der Befehl Alle Haltepunkte löschen".

18.9 Der Befehl Alle Haltepunkte löschen



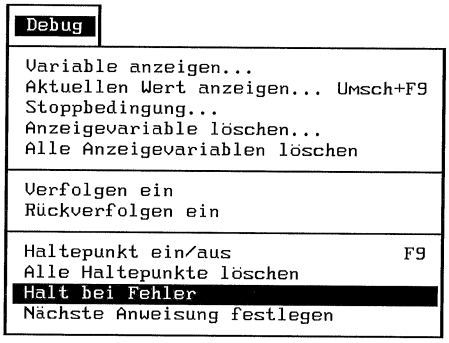
Der Befehl **Alle Haltepunkte löschen** entfernt alle in einem Programm vorhandenen Haltepunkte. Einzelne Haltepunkte lassen sich anhand des Befehls **Haltepunkt ein/aus** entfernen, wobei die restlichen Haltepunkte unverändert beibehalten werden.

Normalerweise wird der Befehl **Alle Haltepunkte löschen** erst dann angewandt, wenn ein Programmabschnitt einwandfrei funktioniert.

Siehe auch

Abschnitt 17.3.2, "Haltepunkte und Stoppbedingungen", und Abschnitt 18.8, "Der Befehl Haltepunkt ein/aus".

18.10 Der Befehl Halt bei Fehler (nur im vollständigen Menü)



Dieser Befehl ermöglicht die Verfolgung einer fehlerverursachenden Anweisung in einem Programm, das über eine Fehlerbehandlungsroutine verfügt. Wenn ein Punkt neben seiner Bezeichnung steht, ist dieser Befehl aktiv und setzt einen impliziten Haltepunkt nach der Marke, die in der zuletzt ausgeführten Fehlerbehandlungsanweisung (**ON ERROR**) angegeben wurde. Der Befehl ist jedoch überflüssig, wenn keine Fehlerbehandlungsroutine verwendet wird.

Beim Auftreten eines Fehlers in einem Programm, das über eine Fehlerbehandlungsroutine verfügt, lenkt QuickBASIC die Ausführung zu der Marke um, die in der entsprechenden **ON ERROR**-Anweisung angegeben ist. Wenn sich keine **ON ERROR**-Anweisung ausdrücklich für den betreffenden Fehlertyp eignet, ist es schwierig, die Fehlerquelle zu finden. Der Befehl **Halt bei Fehler** hilft dieses Problem zu lösen.

Er aktiviert den Befehl **Rückverfolgen ein** und erfaßt die fehlerverursachende Anweisung, die durch Auswahl des Befehls **Verfolgen rückwärts** (Tastenkombination UMSCHALT+F8) angezeigt wird.

Im folgenden Programmabschnitt tritt ein Fehler in der Zeile `Y=5/X` auf. Dieses Beispiel zeigt, wie QuickBASIC anhand des Befehls **Halt bei Fehler** nach der in der letzten `ON ERROR`-Anweisung angegebenen Marke anhält.

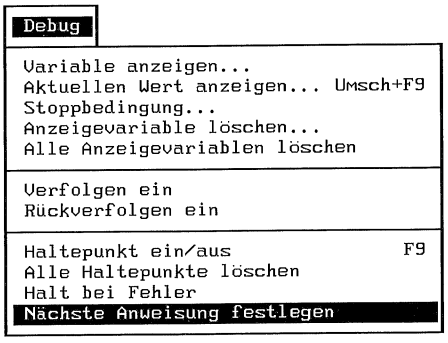
```
ON ERROR GOTO ErstFehlerbehand     'Erste Fehlerbehandlung
ON ERROR GOTO NächstFehlerbehand   'Zweite Fehlerbehandlung
ON ERROR GOTO LetztFehlerbehand    'Dritte und
                                   'letzte
                                   'Fehlerbehandlung
.
X = 0
Y = 5/X
.
LetztFehlerbehand:   'Marke in der zuletzt ausgeführten
                     'Fehlerbehandlungsroutine
.                    'QuickBASIC hält unmittelbar nach der
                     'Marke an
.
```

Wenn der Befehl **Halt bei Fehler** aktiv ist, verursacht ein Fehler die Umlenkung des Programmablaufs zu `LetztFehlerbehand`, und die Ausführung wird unterbrochen. Der Befehl **Rückverfolgen ein** zeigt jetzt die fehlerverursachende Anweisung an.

Siehe auch

Abschnitt 18.7, "Der Befehl Rückverfolgen ein".

18.11 Der Befehl Nächste Anweisung festlegen



Mit Hilfe dieses Befehls kann ein Programmabschnitt übersprungen, wiederholt oder unterbrochen werden.

Der Befehl **Nächste Anweisung festlegen** ändert die Ausführungsfolge, sodaß die an der Cursorstelle stehende Anweisung unabhängig vom dazwischenstehenden Programmcode als nächste Anweisung ausgeführt wird. Bei der Verwendung dieses Befehls ist Vorsicht geboten, da das Überspringen von Codeabschnitten zu undefinierten Variablen und fehlerhaften Ergebnissen führen kann.

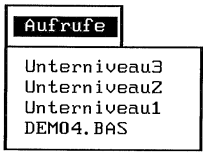
Der Befehl **Nächste Anweisung festlegen** wirkt sich ähnlich wie die Anweisung **GOTO** aus und weist dieselben Einschränkungen auf. Es ist beispielsweise nicht möglich, von der Modul-Ebene eines Programms in die Mitte einer Prozedur zu verzweigen.

Im Unterschied zum Befehl **Nächste Anweisung** aus dem Menü **Ansicht**, der den Cursor zur nächsten bei der Fortsetzung des Programmablaufs auszuführenden Anweisung bringt, erlaubt der Befehl **Nächste Anweisung festlegen** aus dem Menü **Debug** dem Programmierer, die nächste auszuführende Anweisung selbst zu bestimmen. Der Befehl **Nächste Anweisung** gibt die nächste auszuführende Anweisung an, unter der Voraussetzung, daß keine Änderungen vorgenommen werden, während der Befehl **Nächste Anweisung festlegen** die Wahl einer anderen Stelle für die Fortsetzung der Programmausführung ermöglicht.

Siehe auch

Abschnitt 14.4, "Der Befehl Nächste Anweisung".

19 Das Menü Aufrufe (Vollständiges Menü)



Das Menü **Aufrufe** ist eine Einrichtung, die alle zur aktuellen Stelle im Programmablauf führenden Prozeduraufrufe anzeigt. Mit Hilfe dieses Menüs ist der Programmierer in der Lage:

- Die bis zur aktuellen Prozedur ausgeführte Folge verschachtelter Prozeduraufrufe zu überblicken.
- Das Programm von dem Unterbrechungspunkt bis zu einer beliebigen aufgelisteten Prozedur fortzusetzen.

Anhand dieses Menüs kann festgestellt werden, welche Prozeduren bereits ausgeführt wurden. Während der Fehlerbeseitigung kann das Menü **Aufrufe** zusammen mit dem in Abschnitt 17.3.6, "Andere Debug-Einrichtungen", beschriebenen Einzelschrittmodus zur Programmausführung bis zur Fehlerstelle eingesetzt werden, um die Variablenwerte und Bedingungen zu rekonstruieren, welche zum Programmabsturz führten.

Falls Ihre Prozeduren keine weiteren Prozeduren aufrufen, zeigt das Menü **Aufrufe** nur den Namen des Hauptmoduls und die momentan ausgeführte Prozedur an. Vor der Ausführung zeigt das Menü **Aufrufe** in allen Programmen lediglich den Programmnamen an.

19.1 Die Anwendung des Menüs Aufrufe

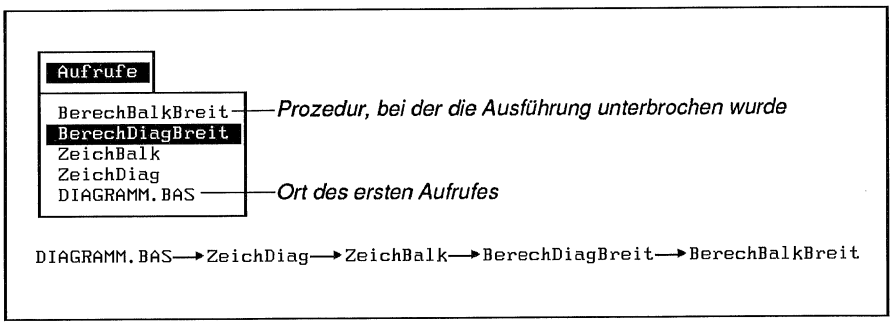
Das Menü **Aufrufe** zeigt die gegenseitigen Aufrufe von Prozeduren an. Nachstehendes Beispiel veranschaulicht die Anwendung dieses Menüs während der Fehlerbeseitigung und der Ausführung.

Nehmen wir beispielsweise an, daß Sie ein Programm zum Zeichnen von Balkendiagrammen schreiben möchten und Sie den Benutzer zur Angabe des Bildschirmteils, in dem das Diagramm erscheinen soll, auffordern wollen. Als Kenner der Vorteile einer strukturierten Programmierung werden Sie den Vorgang in mehrere Prozeduren aufteilen.

Das Hauptmodul fordert den Benutzer auf, die entsprechenden Daten einzugeben und ruft dann eine Prozedur mit dem Namen `ZeichDiag` zum Zeichnen des Balkendiagramms auf. Diese Prozedur übergibt anschließend die Aufgabe, die Balken des Diagramms zu zeichnen, an eine weitere Prozedur, `ZeichBalk`. Da ein Diagramm eine unterschiedliche von den eingegebenen Daten abhängige Anzahl von Balken aufweisen kann, gibt `ZeichBalk` die Berechnung der Breite jedes einzelnen Balken an die Prozedur `BerechBalkBreit` weiter. Bevor jedoch `BerechBalkBreit` die Breite der einzelnen Balken bestimmen kann, muß es die Breite des gesamten Diagramms kennen und ruft zu diesem Zweck die Prozedur `BerechDiagBreit` auf, die die Breite der Diagrammanzeige berechnet.

Während der Ausführung ruft somit `ZeichDiag ZeichBalk` auf, das seinerseits `BerechBalkBreit` aufruft. Letztere ruft dann `BerechDiagBreit` auf. Wenn die Ausführung bei `BerechDiagBreit` unterbrochen wird, erscheinen die Elemente des Menüs **Aufrufe** wie in Abbildung 19.1 dargestellt.

Abbildung 19.1 Folge von Prozeduren im Menü Aufrufe



Das Menü **Aufrufe** zeichnet die "Spur" auf, die die Programmausführung bis zur Prozedur `BerechDiagBreit` aufzeigt. Dies ist besonders nach einer Programmunterbrechung bei vielfach verschachtelten Prozeduren (wie im Fall des Balkendiagramms) hilfreich. Die Reihenfolge der im Menü **Aufrufe** stehenden Punkte entspricht der Reihenfolge der Prozeduraufrufe. Mit anderen Worten bedeutet das, daß die am Anfang des Verzeichnisses stehende Prozedur von der zweiten Prozedur, diese ihrerseits von der dritten, usw. aufgerufen wurde. Demnach wurde das erste Menüelement zuletzt ausgeführt.

Beim Absturz einer Prozedur gibt das Menü **Aufrufe** die Folge der zu diesem Absturz führenden Prozeduren an.

Wenn beispielsweise das vorher beschriebene Programm zum Zeichnen von Diagrammen in der Prozedur `BerechDiagBreit` abstürzt, kann das Menü **Aufrufe** bei Fortsetzung des Programms oder bei der Rekonstruktion der zum Absturz führenden Bedingungen behilflich sein.

Das Programm kann wie folgt mit der Prozedur `BerechDiagBreit` fortgesetzt werden:

1. Ändern Sie die zum Fortsetzen des Programms erforderlichen Variablen im Direktfenster (beseitigen Sie die zum Absturz führenden Fehler).
2. Wählen Sie `BerechBalkBreit` aus dem Menü **Aufrufe** aus, und drücken Sie die EINGABETASTE. Dieses Verfahren bringt die Prozedur `BerechBalkBreit` ins Arbeitsfenster, wobei sich der Cursor unterhalb der Zeile befindet, aus der `BerechDiagBreit` aufgerufen wurde.
3. Betätigen Sie die Taste F7, um das Programm vom aktuellen Punkt der Unterbrechung (die Anweisung, an der der Absturz erfolgte) bis zur aktuellen Cursorstelle (unmittelbar nach dem Aufruf der Prozedur `BerechDiagBreit`) auszuführen. Der restliche Code der Prozedur `BerechDiagBreite` wird ausgeführt und das Programm an der Cursorstelle in der Prozedur `BerechBalkBreit` angehalten.
4. Überprüfen Sie mit Hilfe des Debug-Fensters, des Direktfensters oder des Befehls **Aktuellen Wert anzeigen**, ob die Variablen die gewünschten Werte aufweisen, oder prüfen Sie die Ausgabe auf Änderungen anhand des Befehls **Ausgabebildschirm**. Beide Methoden bieten die Möglichkeit, die Funktionstüchtigkeit des Programms nach den im 1. Arbeitsschritt vorgenommenen Änderungen zu überprüfen.

Die zum Absturz führenden Bedingungen lassen sich wie folgt rekonstruieren:

1. Bringen Sie den Cursor an den Anfang des Programms, und wählen Sie den zum Menü **Debug** gehörenden Befehl **Nächste Anweisung festlegen** aus, um die Ausführung des Programms vorzubereiten.
2. Wählen Sie `BerechBalkBreit` im Menü **Aufrufe** aus, und drücken Sie die EINGABETASTE. QuickBASIC zeigt die Prozedur `BerechBalkBreit` an, wobei sich der Cursor auf der Zeile unterhalb des Aufrufes von `BerechDiagBreit` befindet. Bewegen Sie den Cursor in die Zeile oberhalb dieses Aufrufes.
3. Betätigen Sie die Taste F7, um das Programm von der aktuellen Stelle der Ausführung (der ersten ausführbaren Anweisung im Programm) bis zu der aktuellen Cursorstelle auszuführen.
4. Führen Sie das Programm von `BerechBalkBreit` zu `BerechDiagBreit` durch Betätigen der Taste F8 schrittweise aus, um die fehlerverursachende Anweisung zu finden.
5. Beseitigen Sie den Fehler mit Hilfe des Befehls **Aktuellen Wert anzeigen** oder anderer Debug-Techniken.

Auf diese Weise kann das Menü **Aufrufe** die Fehlerbeseitigung in einem komplexen Programm vereinfachen.

19.2 Aktive Prozeduren

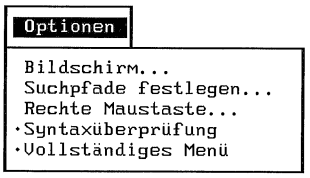
Die aktuell aktive Prozedur wird an erster Stelle im Menü **Aufrufe** aufgelistet. Unmittelbar darunter steht die Prozedur, die die aktive Prozedur aufgerufen hat und an die QuickBASIC beim Verlassen der aktiven Prozedur die Kontrolle zurückgibt. Durch Auswahl eines beliebigen Prozedurnamens aus dem Menü **Aufrufe** und Betätigen der Taste F7 läßt sich das Programm bis zu der betreffenden Prozedur ausführen.

Gehen wir beispielsweise von den in Abbildung 19.1 aufgeführten Prozeduren aus. Da `BerechBalkBreit` hervorgehoben ist wird durch Drücken der EINGABETASTE und der Taste F7 der Code vom Anfang des Programms bis zu der dem Aufruf von `BerechDiagBreit` folgenden Anweisung in `BerechBalkBreit` ausgeführt. Es ist dabei zu beachten, daß das Menü **Aufrufe** nur die letzte Ausführung verfolgt, so daß jetzt die Prozedur `BerechBalkBreit` als Erste im Menü stehen würde, nachdem `BerechDiagBreit` entfernt wurde, da sie nicht zum zuletzt ausgeführten Programmabschnitt gehört.

Das Menü **Aufrufe** kann als "Stapel" bezeichnet werden, da es die zuletzt aufgerufene Prozedur als Erste aufführt. Das Verzeichnis dieses Menüs umfaßt nur die letzten 8 zuletzt aufgerufenen Prozeduren. Weitere Verschachtelungen werden nicht angezeigt, bis die Folge der Prozeduraufrufe durch Fortsetzen der Ausführung zurückverfolgt wird.

Die Eigenschaften des Aufrufstapels können anhand der am Ende des Abschnitts 18.1, "Der Befehl Variable anzeigen", beschriebenen Fibonacci-Folge eingehend untersucht werden. Dank der rekursiven Prozedur `Fib` bestimmt die nach der Eingabeaufforderung eingegebene Zahl die Verschachtelungstiefe des Aufrufstapels sowie die Anzahl der im Menü **Aufrufe** vorhandenen Prozeduren. Nach Unterbrechung der Ausführung durch eine Stoppbedingung oder durch Betätigen der Tastenkombination STRG+UNTBR können Sie anhand des Menüs **Aufrufe** die verschiedenen Wiederholungen der Prozedur `Fib` mit oben erklärten Methoden untersuchen. Da es sich um eine rekursive Prozedur handelt, sind alle Prozedurnamen identisch, und sämtliche aufgeführte Prozeduren stellen einen weiteren Aufruf derselben Prozedur dar.

20 Das Menü Optionen



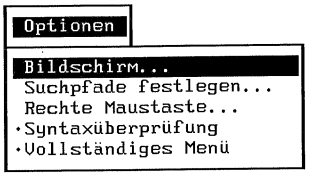
Das Menü Optionen ermöglicht es, den QuickBASIC-Bildschirm den eigenen Bedürfnissen anzupassen, Standard-Suchpfade festzulegen, die Funktion der rechten Maustaste zu bestimmen, die Syntaxüberprüfung ein- und auszuschalten oder zwischen dem vollständigen Menü und dem Kurzmenü zu wechseln.

Es umfaßt folgende Befehle:

- **Bildschirm**. Paßt die QuickBASIC-Umgebung den eigenen Bedürfnissen an (Farbe, Bildlaufleisten, Tabulatorstopps).
- **Suchpfade festlegen**. Legt die für bestimmte Dateitypen geeigneten Suchpfade fest.
- **Rechte Maustaste**. Ändert die Funktion der rechten Maustaste (nur im vollständigen Menü).
- **Syntaxüberprüfung**. Schaltet die Syntaxüberprüfung ein oder aus (nur im vollständigen Menü).
- **Vollständiges Menü**. Schaltet das vollständige Menü ein oder aus.

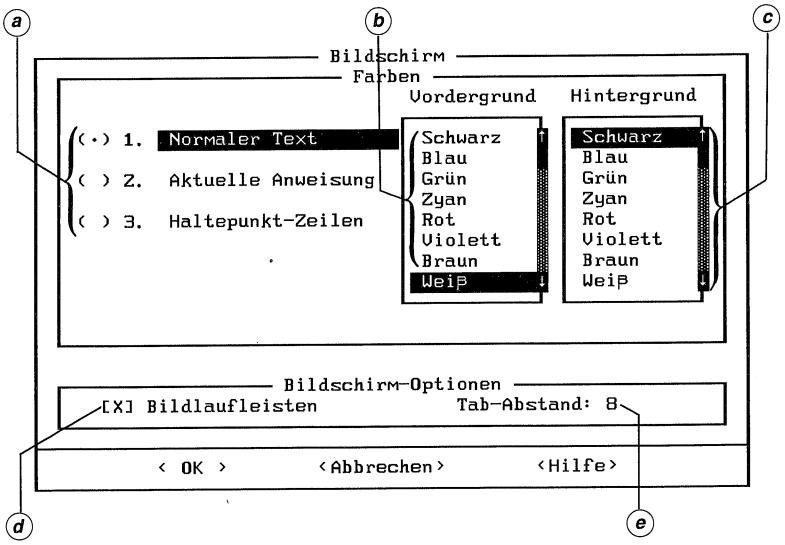
Beim Ändern der Umgebung durch Starten von QuickBASIC anhand einer Befehlszeilenoption (wie in Abschnitt 10.1.1, "Der QB-Befehl", beschrieben) oder durch Anwenden des Menüs **Optionen** speichert QuickBASIC die vorgenommenen Änderungen in einer Datei mit dem Namen *qb.ini*. Wenn die Datei bei der Änderung bereits besteht, wird sie aktualisiert, andernfalls wird sie zu diesem Zeitpunkt angelegt. Wenn die Datei *qb.ini* nicht existiert, verwendet QuickBASIC die Standard-Optionen.

20.1 Der Befehl Bildschirm



Bei Auswahl dieses Befehls erscheint das Dialogfeld **Bildschirm**, in dem Hinter- und Vordergrundfarben sowie Tabulator-Abstände (Anzahl der Leerstellen, um die der Cursor durch Betätigen der TAB-TASTE im Arbeitsfenster eingerückt wird) festgelegt werden können. In diesem Dialogfeld können Sie ebenfalls bestimmen, ob Bildlaufleisten im Arbeitsfenster erscheinen sollen oder nicht (siehe Abbildung 20.1).

Abbildung 20.1 Das Dialogfeld Bildschirm



a) *Textart, die geändert werden soll.*
b) *Farboptionen für den Vordergrundtext.*
c) *Farboptionen für den Hintergrund.*
d) *Schaltet bei Auswahl die Bildlaufleisten ein.*
e) *Anzahl der durch Betätigen der Tab-Taste eingerückten Leerstellen.*

Die Hinter- und Vordergrundfarben lassen sich wie folgt ändern:

1. Bringen Sie den Cursor in die Klammern, die vor der zu ändernden Farbeneinstellung stehen (Normaler Text, Aktuelle Anweisung oder Haltepunkt-Zeilen).
2. Drücken Sie die TAB-TASTE, um den Cursor zum Verzeichnisfeld **Vordergrundfarbe** zu bewegen. Wählen Sie die gewünschte Vordergrundfarbe für das im 1. Arbeitsschritt ausgewählte Element durch betätigen der RICHTUNGSTASTEN aus. Das zuvor ausgewählte Element zeigt die geänderten Farben an.
3. Betätigen Sie erneut die TAB-TASTE, um sich zum Verzeichnisfeld **Hintergrundfarbe** zu bewegen, und wiederholen Sie das im vorhergehenden Arbeitsschritt beschriebene Verfahren zum Ändern der Hintergrundfarbe.
4. Drücken Sie die EINGABETASTE.

Die QuickBASIC-Umgebung erscheint in den neuen Farben.

Das zum Dialogfeld **Bildschirm** gehörende Feld **Bildschirm-Optionen** steuert sowohl Bildlaufleisten als auch die Einstellung der Tabulator-Abstände. Wenn sich der Cursor im Kontrollfeld **Bildlaufleisten** befindet, können die Bildlaufleisten im Arbeitsfenster durch Betätigen der LEERTASTE ein- oder ausgeschaltet werden (diese Option ist standardmäßig eingeschaltet). Die Option **Tab-Abstand** steuert die Anzahl der Leerstellen, um die der Cursor durch Betätigen der TAB-TASTE eingerückt wird. Benutzen Sie einen geeigneten Abstand um Ihre Programme lesbar zu gestalten.

20.2 Der Befehl Suchpfade festlegen

Der Befehl **Suchpfade festlegen** ändert die Standard-Suchpfade für bestimmte Dateitypen.

Es kann vorkommen, daß Sie sämtliche Dateien eines bestimmten Typs zwecks besserer Organisation in einem eigenen Verzeichnis speichern möchten. Es ist zum Beispiel ratsam, daß sich sämtliche QuickBASIC- sowie die eigenen Quick-Bibliotheken in einem oder mehreren Unterverzeichnissen der QuickBASIC-Diskette oder des QuickBASIC-Verzeichnisses befinden. Da die Umgebung Zugriff auf die entsprechenden Bibliotheken erfordert, müßten Sie normalerweise einen Pfadnamen für jede Bibliothek angeben.

Im Dialogfeld **Suchpfade festlegen** können Sie einen Suchpfad, der bis zu 127 Zeichen betragen kann, für die Dateien eingeben. Dieses Verfahren erweist sich beim Zugriff auf Bibliotheken und andere nicht im aktuellen Arbeitsverzeichnis vorhandene Dateien als zeitsparend, da Sie nicht jedes Mal den vollständigen Pfadnamen einzugeben brauchen.

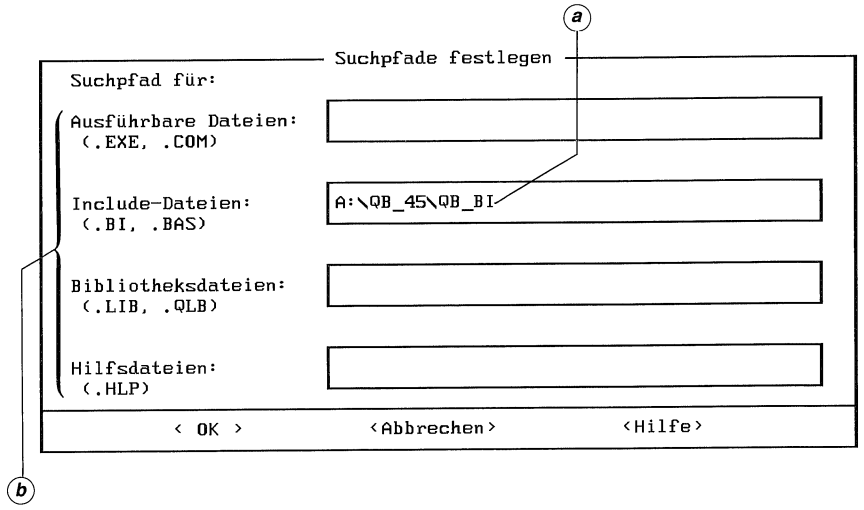
Wenn beispielsweise alle Include-Dateien und Programme in einem Verzeichnis mit dem Namen `QB_BI` gespeichert sind, das sich seinerseits im Verzeichnis mit dem Namen `QB_45` auf einer Diskette im Laufwerk `A` befindet, können Sie QuickBASIC wie folgt anweisen, nach Dateien, die mit der Erweiterung *.bi* oder *.bas* versehen sind, in dem entsprechenden Verzeichnis zu suchen:

1. Wählen Sie den Befehl **Suchpfade festlegen** aus dem Menü **Optionen**. Das Dialogfeld **Suchpfade festlegen** erscheint.
2. Bringen Sie den Cursor durch Betätigen der TAB-TASTE in das Textfeld **Include-Dateien**.
3. Geben Sie den Pfadnamen ein:

   ```
   A: \QB_45\QB_BI
   ```

   Das Dialogfeld wird in Abbildung 20.2 dargestellt.

Abbildung 20.2 Das Dialogfeld Suchpfade festlegen

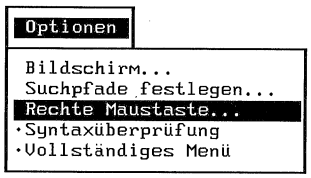


a) Der zum Verzeichnis mit den Dateien führende Pfad.
b) Dateitypen.

4. Drücken Sie die EINGABETASTE.

Hinweis Die im Dialogfeld **Suchpfade festlegen** eingegebenen Suchpfade ändern nicht die auf DOS-Ebene mit den Befehlen PATH und SET festgelegten Pfade oder Umgebungsvariablen. Die mit dem Befehl **Suchpfade festlegen** gesetzten Pfade kommen nur innerhalb von QuickBASIC zur Anwendung.

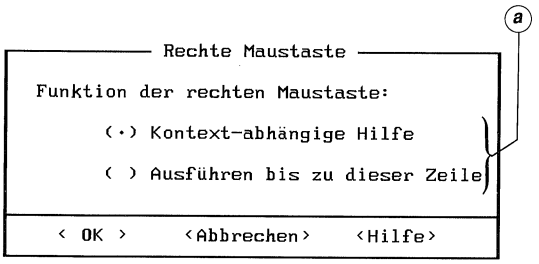
20.3 Der Befehl Rechte Maustaste (nur im vollständigen Menü)



Mit Hilfe dieses Befehls läßt sich die Funktion der rechten Maustaste ändern.

Bei Auswahl des Befehls **Rechte Maustaste** erscheint das in Abbildung 20.3 dargestellte Dialogfeld.

Abbildung 20.3 Das Dialogfeld Rechte Maustaste



a) Funktionen der rechten Maustaste.

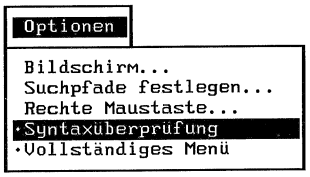
Dieses Dialogfeld bietet die Wahl zwischen zwei Funktionen der rechten Maustaste:

- Aufruf der On-Line-Hilfe, (dieselbe Funktion wie der Befehl **Begriff** aus dem Menü **Hilfe**), der die Standard-Einstellung darstellt.
- Ausführung des Programms vom Anfang bis zur aktuellen Cursorstelle.

Die unterschiedlichen Vorteile beider Funktionen sind in verschiedenen Situationen ersichtlich. Beim Schreiben eines Programms ist der Hilfeaufruf besonders für den Programmieranfänger von Nutzen. Während der Fehlerbeseitigung, dagegen, ist die Programmausführung wichtiger.

Bitte beachten Sie bei der Verwendung der Option **Kontextabhängige Hilfe**, daß sich die auf dem Bildschirm sichtbare Hilfe auf das Wort bezieht, auf dem der Cursor steht.

20.4 Der Befehl Syntaxüberprüfung (nur im vollständigen Menü)



Dieser zum Menü **Optionen** gehörende Befehl schaltet die Syntaxüberprüfung ein oder aus.

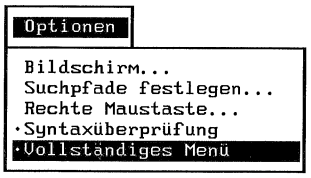
Diese Option ist standardmäßig eingeschaltet, was durch einen neben der Bezeichnung des Befehls stehenden Punkt (•) angezeigt wird. Wählen Sie den Befehl aus, um die Syntaxüberprüfung auszuschalten. Der Punkt verschwindet. Da die ausgewählten Optionen beim Verlassen von QuickBASIC gespeichert werden, bleibt die Syntaxüberprüfung beim nächsten Aufruf von QuickBASIC ausgeschaltet.

Wenn sie eingeschaltet ist, überprüft diese Einrichtung jede Zeile unmittelbar nach Eingabe auf die richtige Syntax.

Siehe auch

Abschnitt 12.5.2, "Die automatische Syntaxüberprüfung".

20.5 Der Befehl Vollständiges Menü



Der Befehl **Vollständiges Menü** wechselt zwischen dem erweiterten Menü von QuickBASIC und dem Kurzmenü. Wenn der Befehl eingeschaltet ist, erscheint ein Punkt (•) neben dessen Bezeichnung.

Die QuickBASIC-Menüs können in ihrer "kurzen" oder "vollständigen" Form verwendet werden. Die Kurzmenüs enthalten alle zum Erstellen einfacher und mittlerer Programme erforderlichen Funktionen, umfassen jedoch keine Befehle für den fortgeschrittenen Programmierer, um den Gebrauch der Umgebung für Programmieranfänger zu vereinfachen. In ihrer vollständigen Form enthalten die Menüs ein zusätzliches Element, das Menü **Aufrufe**, sowie weitere Befehle innerhalb der einzelnen Menüs. Verwenden Sie die Option **Vollständiges Menü**, wenn Ihnen QuickBASIC vertraut ist und Sie die weitere Funktionen benötigen.

Beim Verlassen von QuickBASIC werden alle ausgewählten Optionen, einschließlich **Bildschirm**, **Suchpfade**, **Syntaxüberprüfung** und **Vollständiges Menü** gespeichert und sind beim nächsten Start von QuickBASIC weiterhin aktiv.

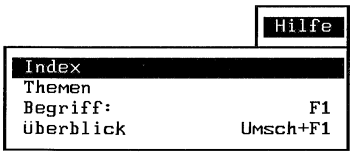
21 Das Menü Hilfe

Die Befehle dieses Menüs bieten eine Möglichkeit, auf die On-Line-Hilfe von QuickBASIC zuzugreifen. (Nähere Informationen zu weiteren Zugriffsmöglichkeiten auf die On-Line-Hilfe sind Abschnitt 10.8, "Verwendung der On-Line-Hilfe", zu entnehmen.)

Das Menü **Hilfe** verfügt über folgende Befehle:

- **Index**. Zeigt alphabetisches Verzeichnis der QuickBASIC-Schlüsselwörter und eine Kurzbeschreibung der einzelnen Schlüsselwörter an.
- **Themen**. Bietet einen Überblick der in der On-Line-Hilfe behandelten Themen.
- **Begriff**. Erteilt Informationen über Syntax und Anwendung gegebener Variablen und QuickBASIC-Schlüsselwörter.
- **Überblick**. Erläutert die Verwendung der On-Line-Hilfe und der Tastenkombinationen zur direkten Befehlseingabe.

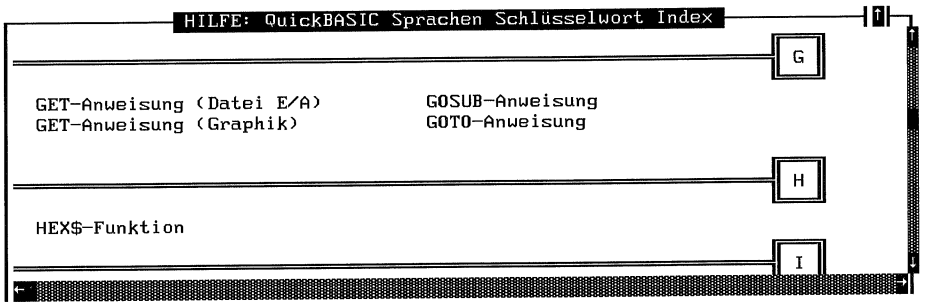
21.1 Der Befehl Index



Dieser Befehl liefert ein alphabetisches Verzeichnis der QuickBASIC-Schlüsselwörter. Zum Anfordern von Informationen zu einem bestimmten Schlüsselwort ist der Cursor in die Zeile mit dem betreffenden Schlüsselwort zu bringen und die Taste F1 zu betätigen. Anschließend erscheint die entsprechende Hilfe für das Schlüsselwort.

Sie können das Schlüsselwortverzeichnis mit den RICHTUNGSTASTEN durchgehen oder direkt zu einem bestimmten Buchstaben der Liste springen. Nach Eingabe des Buchstabens G, bewegt sich der Cursor zum Beispiel zu GET, dem ersten Schlüsselwort, das mit einem "G" beginnt (siehe Abbildung 21.1).

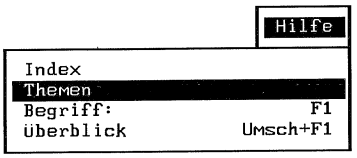
Abbildung 21.1 Index-Einträge im Menü Hilfe



Siehe auch

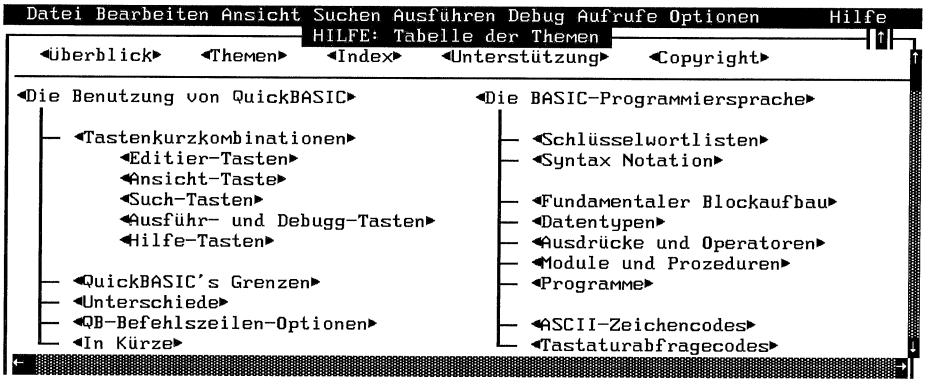
Abschnitt 21.3, "Der Befehl Begriff".

21.2 Der Befehl Themen



Der Befehl **Themen** zeigt ein Inhaltsverzeichnis der On-Line-Hilfe an. Er bietet auch Hinweise zur Anwendung der QuickBASIC-Umgebung und der BASIC-Sprache sowie nützliche Tabellen. Sie können aus diesem Inhaltsverzeichnis einfach zum gewünschten Thema übergehen (siehe Abbildung 21.2).

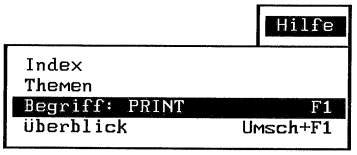
Abbildung 21.2 Inhaltsverzeichnis der On-Line-Hilfe



Positionieren Sie den Cursor auf das gewünschte Thema, und drücken Sie die Taste F1, um ausführliche Informationen zum betreffenden Thema zu erhalten.

Das Fenster **Inhalt** weist beispielsweise unter "Die Benutzung von QuickBASIC" mehrere Einträge auf. Wird der Cursor auf "Bearbeiten" positioniert und die Taste F1 betätigt, erscheint eine Liste der am häufigsten beim Bearbeiten verwendeten Tastenkombinationen zur direkten Befehlseingabe.

21.3 Der Befehl Begriff



Der Befehl **Begriff** erteilt spezifische Informationen über Syntax und Anwendung der QuickBASIC-Variablen, Symbole, Schlüsselwörter, Menüs, Befehle oder Dialogfelder, die sich an der Cursorstelle befinden. Wenn dieser Befehl aktiv ist, erscheint das Thema der On-Line-Hilfe nach dem Wort `Begriff` im Menü **Hilfe**. Der Befehl **Begriff** wirkt sich wie die Betätigung der Taste F1 aus.

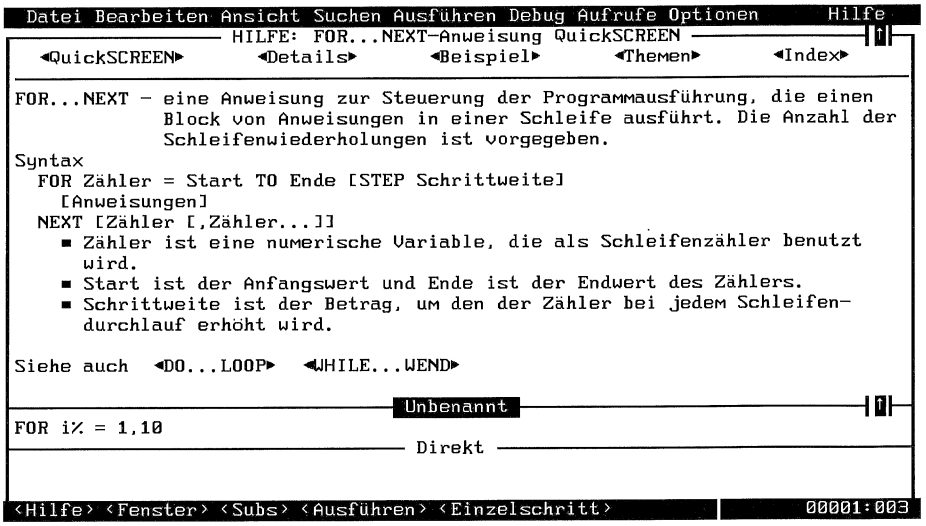
Durch Eingabe des Wortes und Auswahl des Befehls **Begriff** können Sie jederzeit Hilfe für ein beliebiges Schlüsselwort oder eine Variable erhalten.

Wird dieser Befehl beim Antreffen eines Syntaxfehlers ausgewählt, kann die Anwendung der betreffenden Anweisung sofort verbessert werden. Wenn Sie beispielsweise die nachstehende Zeile eingeben, in der ein Komma anstelle des Wortes `TO` steht:

```
FOR i% = 1, 10
```

markiert QuickBASIC den Syntaxfehler mit der Meldung `Erwartet: TO`. Drücken Sie die Taste F1 oder die Hilfsfläche, um Hilfe für den Fehler zu erhalten. Fordern Sie Informationen über das Schlüsselwort `FOR` an, indem Sie den Cursor zu der Anweisung `FOR` bringen und den Befehl **Begriff** auswählen. Der Programmcode bleibt während dieses Vorgangs am unteren Ende des Bildschirms sichtbar (siehe Abbildung 21.3).

Abbildung 21.3 Hilfe für die Anweisung FOR...NEXT



Beispielcode aus der Hilfe für `FOR` läßt sich wie folgt in das eigene Programm kopieren:

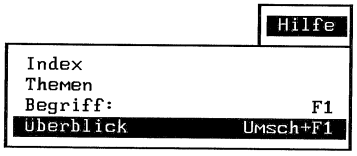
1. Wählen Sie die Hyperverbindung `Beispiel` am oberen Ende des Fensters aus.
2. Markieren Sie den Beispielcode, und führen Sie den Befehl **Kopieren** aus dem Menü **Bearbeiten** aus.
3. Schließen Sie das Hilfsfenster mit der ESC-TASTE.
4. Bringen Sie den Cursor an die Programmstelle, an die Sie den Code kopieren wollen, und führen Sie den Befehl **Einfügen** aus dem Menü **Bearbeiten** aus.

Der eingefügte Code kann nun dem Programm angepaßt oder unverändert benutzt werden.

Taste zur direkten Befehlseingabe

F1

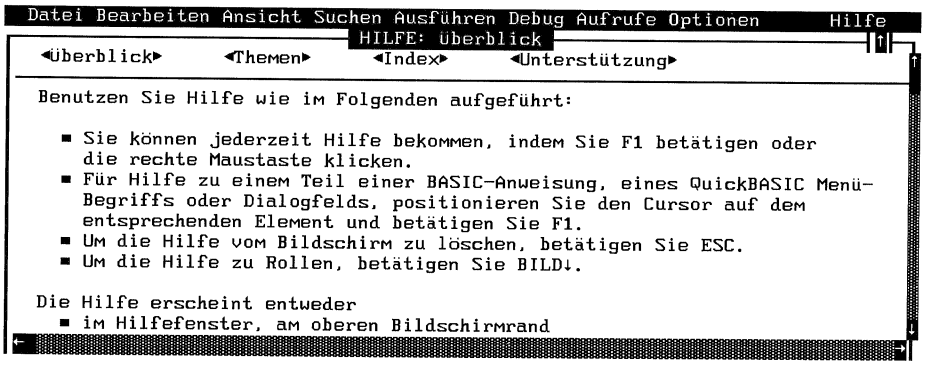
21.4 Der Befehl Überblick



Dieser Befehl erläutert die Verwendung der On-Line-Hilfe von QuickBASIC.

Überblick beschreibt die wichtigsten Einrichtungen der On-Line-Hilfe. Um ausführlichere Informationen zu erhalten, verlassen Sie die Anzeige **Überblick** mit der ESC-TASTE, und führen Sie den Befehl **Begriff** aus dem Menü **Hilfe** aus, oder drücken Sie die Taste F1.

Abbildung 21.4 Die Anzeige Überblick der On-Line-Hilfe



Direkte Tastenkombination

UMSCHALT+F1

Glossar

Ziel dieses Glossars ist es, die in allen Handbüchern zu QuickBASIC, Version 4.5, verwendeten Begriffe eindeutig zu definieren. Weder die einzelnen Definitionen, noch die Liste der Begriffe stellen ein umfassendes Wörterbuch für die Terminologie der Informatik dar.

Absolute Koordinaten Horizontale und vertikale Abstände, die sich auf die obere linke Ecke des Bildschirms mit den Koordinaten (0, 0) beziehen. Absolute Koordinaten werden in Bildpunkten gemessen (siehe auch "Physikalische Koordinaten").

Adresse Zahl zur eindeutigen Kennzeichnung einer Speicherzelle.

Aktiviertes Ereignis Ein Ereignis, nach dem QuickBASIC aktiv sucht und das mit Hilfe von **ON** *Ereignis*-Anweisungen aktiviert wird.

Aktuelles Laufwerk Das Laufwerk mit der Diskette, in der DOS standardmäßig nach einem Verzeichnis oder einer Datei sucht.

Aktuelles Verzeichnis Das Verzeichnis, in dem DOS standardmäßig nach den angegebenen Dateien sucht.

Algorithmus Das schrittweise Vorgehen zum Lösen eines Problems oder Berechnen eines Ergebnisses.

Animation Eine Debug-Einrichtung von QuickBASIC, die jede Zeile eines laufenden Programms während der Ausführung hervorhebt. Die Animation wird durch den Befehl **Verfolgen ein** aus dem Menü **Debug** eingeschaltet.

Anschluß Eine elektrische Verbindung, die die Datenübertragung vom Computer zu anderen Geräten und Anlagen gewährleistet.

Anweisung Eine Kombination von einem oder mehreren BASIC-Schlüsselwörtern, Variablen und Operatoren, die vom Compiler oder Interpreter als eine Einheit behandelt wird. Eine BASIC-Anweisung kann selbständig auf einer einzelnen Programmzeile stehen. Der Begriff Anweisung bezieht sich sowohl auf einzeilige Anweisungen, wie zum Beispiel **OPEN** oder **INPUT**, als auch auf eine gesamte, aus mehreren Zeilen bestehende Anweisung, wie **IF...END IF** oder **SELECT CASE...END SELECT**. Bestandteile zusammengesetzter Anweisungen, wie zum Beispiel **ELSE** oder **CASE**, werden als "Klauseln" bezeichnet.

Anzeigbar Ein Debug-Begriff, der angibt, ob eine Variable oder ein Ausdruck im aktuellen Kontext der Ausführung beobachtet werden kann. Ein Element ist nur während der Ausführung des Programmteils anzeigbar, aus dem das betreffende Element im Debug-Fenster eingesetzt wurde.

Arbeitsfenster Das Haupt-Editierfenster, in das auch Programme geladen werden.

Argument Ein Datenelement, das einer BASIC-Anweisung, -Funktion, -Prozedur oder einem Programm übergeben wird. Im nachfolgenden Beispiel ist die Zeichenkette `"Hallo"` ein Argument der BASIC-Anweisung `PRINT: PRINT "Hallo"`. Sowohl Konstanten als auch Variablen können Argumente darstellen, die häufig zur Übergabe von Informationen von einem Programmteil an einen anderen verwendet werden. Argumente der DOS-Befehlszeile können dem Programm mit Hilfe der Funktion **COMMAND$** zur Verfügung gestellt werden.

Arithmetische Umwandlung Umwandlung von Ganzzahlen in Gleitkommazahlen und umgekehrt.

ASCII (American Standard Code for Information Interchange) Ein Satz von 256 Zeichencodes, der von zahlreichen Computern zur Darstellung von Buchstaben, Zahlen, Sonderzeichen und anderen Symbolen verwendet wird. 128 Zeichencodes sind normiert, während die restlichen 128 Codes Sonderzeichen darstellen, die vom Computerhersteller definiert werden.

Attribute Farbe, Helligkeit, Blinken und andere angezeigte Eigenschaften von Zeichen oder Bildpunkten.

Ausdruck Eine Kombination von Operanden und Operatoren, die einen einzelnen Wert ergibt.

Ausführbare Datei Eine mit der Erweiterung *.exe, .com* oder *.bat* versehene Datei, die durch Eingabe des Dateinamens auf der DOS-Befehlszeile gestartet werden kann.

Ausführen Die Ausführung eines Programms oder einer bestimmten Anweisung im Direktfenster.

Ausgabe Alle Daten, die vom Prozeßrechensystem ausgegeben werden und normalerweise auf Eingabedaten basieren.

Ausgabebildschirm Der Bildschirm, in dem die Ausgabe bei der Ausführung eines Programms in der QuickBASIC-Umgebung erscheint. Nach der Fehlerbeseitigung bleibt der Ausgabebildschirm bei Ausführung des Programms außerhalb der QuickBASIC-Umgebung unverändert.

Auswählen Aussuchen einzelner Elemente, wie zum Beispiel Optionsflächen, in einem Dialogfeld. Dieses Verb bezeichnet auch die Hervorhebung von Textabschnitten oder Graphik zum Bearbeiten.

Automatische Variable Eine zu einer **SUB**- oder **FUNCTION**-Prozedur gehörende Variable, deren Wert zwischen den einzelnen Aufrufen einer Prozedur nicht gespeichert wird. Automatische Variablen werden häufig in rekursiven Prozeduren, deren Werte zwischen den Aufrufen nicht gesichert werden brauchen, verwendet.

Basisname Der vor der Erweiterung stehende Teil eines Dateinamens. *demo* ist beispielsweise der Basisname der Datei *demo.bas*.

Benutzerdefinierter Typ Eine zusammengesetzte Datenstruktur, die sowohl Zeichenketten als auch numerische Variablen enthalten kann und den Strukturen einer C-Sprache oder den PASCAL-Records ähnlich ist. Benutzerdefinierte Typen werden mit **TYPE**-Anweisungen definiert, während die Datenstruktur mit der Anweisung **TYPE...END TYPE** definiert werden.

Bildlaufleisten Die Leisten, die auf der rechten Seite sowie im unteren Teil des Arbeitsfensters und einiger Verzeichnisfelder erscheinen. Die Bildlaufleisten erlauben die Bewegung des Inhalts eines Arbeitsfensters oder eines Textfeldes mit Hilfe der Maus.

Binär Ein mathematischer Begriff, der sich auf das Zahlensystem mit der Basis zwei bezieht. Es werden nur die Werte 0 und 1 benutzt.

Binäre Datei Bezeichnet in Ein- bzw. Ausgabeoperationen Dateien oder Daten, die sowohl nicht-ausgebbare als auch ASCII-Zeichen enthalten. Der Zugriff auf eine binäre Datei ist die Fähigkeit, auf ein beliebiges Byte innerhalb einer Datei durch dessen numerischen Wert zuzugreifen.

Binden Lösen der Bezüge, die zwischen den Modulen eines Programms sowie dem Programm und den Bibliotheksroutinen bestehen, und Erstellen eines vollständig funktionellen Programms.

BIOS (Basic Input/Output System) In Maschinensprache geschriebene Programme. Diese Programme sind Teil des Betriebssystems und stellen die in einem Computersystem elementaren Operationen zur Verfügung. Die BIOS-Programme können direkt mit den Anweisungen **INTERRUPT** und **CALL INTERRUPTX** benutzt werden.

Bit Die kleinste im Computer verwendete Informationseinheit, die einem Binärzeichen (0 oder 1) entspricht. Acht Bits bilden ein Byte.

Block Eine Folge von Vereinbarungen, Definitionen oder Anweisungen.

Byte Eine für den Speicher des Computers und das Speichern von Daten verwendete Maßeinheit. Ein Byte besteht aus 8 Bits und kann ein beliebiges Zeichen (einen Buchstaben, eine Zahl, ein Satzzeichen oder ein anderes Symbol) darstellen.

CGA (Color Graphics Adapter) Ein Bildschirmadapter, der in der Lage ist, sowohl Textzeichen als auch Graphikbildpunkte anzuzeigen. Mit einem geeigneten Monitor ist auch Farbanzeige möglich.

Compiler Ein Programm, das Anweisungen in eine für den Computer verständliche Sprache übersetzt.

Cursor Die blinkende Linie oder das blinkende Rechteck, das auf dem Bildschirm die Stelle anzeigt, an der das nächste über die Tastatur eingegebene Zeichen dargestellt wird.

Dateiangabe Die vollständige Angabe einer Datei, die den Buchstaben des Laufwerks, den Pfadnamen, den Dateinamen und die Erweiterung umfassen kann.

Dateipuffer Ein Speicherbereich, der Daten vorübergehend aufnimmt. QuickBASIC behält eine Kopie der Datei im Puffer und ändert diese während der Bearbeitung. Beim Speichern der Datei wird der Puffer auf Diskette kopiert.

Daten Vom Computer verarbeitete Zahlen und Text.

Daten-Datei Eine Datei, die vom Programm verwendete oder erstellte Daten, in Form von Text, Zahlen oder als Kombination von beiden, enthält.

Datenfeld Aneinanderreihung von Elementen desselben Typs.

Datenfeld-Grenzen Die oberen und unteren Begrenzungen eines Datenfeldes.

Datensatz Eine Reihe von Feldern, die in einer Datendatei als sinnvolle Einheit behandelt wird, oder eine mit einem benutzerdefinierten Typ deklarierte Variable. Siehe "Benutzerdefinierter Typ".

Datenübertragungsanschluß Siehe "Anschluß".

Debug-Fenster Ein Fenster, das Informationen über die während der Programmausführung überwachten Variablen oder Ausdrücke erteilt. Das Debug-Fenster gibt für das jeweils überwachte Element dessen Speicherplatz, Namen und aktuellen Wert an, wenn das Element anzeigbar ist.

Debugger Ein Programm, das beim Auffinden der während eines Laufzeittests aufgetretenen Fehler behilflich ist.

DEF FN-Funktion Ein mit einer einzeiligen **DEF FN**-Anweisung oder einem **DEF FN...END DEF**-Block definiertes Objekt. (Im allgemeinen wird der **DEF FN**-Funktion eine **FUNCTION**-Prozedur wegen ihrer Austauschbarkeit und Modularität vorgezogen.)

DEFtyp Eine in diesen Handbüchern für die Gruppe von BASIC-Anweisungen verwendete Bezeichnung, die den Standard-Variablentyp eines Programms neu definiert (**DEFDBL**, **DEFINT**, **DEFLNG**, **DEFSNG**, **DEFSTR**).

Dialogfeld Ein Feld, das bei Auswahl eines Befehls erscheint, wenn dieser zusätzliche Informationen erfordert.

Dimension Die Anzahl der erforderlichen Indizes, um ein einzelnes Datenfeldelement anzugeben. Die Anweisung `DIM A$ (4, 12)` deklariert beispielsweise ein zweidimensionales Datenfeld, während die Anweisung `DIM B$ (16, 4, 4)` ein dreidimensionales Datenfeld deklariert.

Direktfenster Der Bereich im unteren Teil des QuickBASIC-Bildschirms, der zur Eingabe und direkten Ausführung von Anweisungen dient.

Direktzugriffsdatei Eine Datei, die in beliebiger Reihenfolge gelesen oder geschrieben werden kann. Siehe "Sequentielle Datei".

Dokument Eine Möglichkeit zur Klassifizierung einer Datei beim Laden in den QuickBASIC-Editor. Das Laden einer Datei als Dokument schaltet die Syntaxüberprüfung, die Großschreibung von Schlüsselwörtern sowie andere Eigenschaften des intelligenten Editors aus und läßt ein BASIC-Programm bei der Bearbeitung eines Dokuments wie ein Textverarbeitungssystem funktionieren.

Doppelte Genauigkeit Ein reeller (Gleitkomma)-Wert, der acht Bytes im Speicher belegt. Werte mit doppelter Genauigkeit sind auf 15 oder 16 Ziffern genau.

Dynamisches Datenfeld Ein Datenfeld, dem der Speicherplatz während der Programmausführung zugewiesen wird. Der Speicherplatz ist nicht vorgegeben und kann während der Programmausführung geändert werden.

E/A-Umlenkung Siehe "Umlenken".

EGA (Enhanced Graphics Adapter) Ein Bildschirmadapter mit der Fähigkeit, in allen Modi des Farbgraphikadapters (CGA) sowie in zahlreichen zusätzlichen Modi zu arbeiten. EGA unterstützt sowohl Text als auch Graphik in mittlerer Auflösung und bis zu 64 Farben.

Einfache Genauigkeit Ein reeller (Gleitkomma-)Wert, der vier Bytes im Speicher belegt. Werte einfacher Genauigkeit sind auf sieben Dezimalstellen genau.

Einfache Variable Eine BASIC-Variable, die kein Datenfeld darstellt. Einfache Variablen können vom Typ numerisch, Zeichenkette oder benutzerdefiniert sein.

Einfüge-Modus Ein Modus, der für die Eingabe von Daten in der QuickBASIC-Umgebung bestimmt ist. In diesem Modus können neue Zeichen eingefügt werden, die jedoch die bereits in der Datei bestehenden Zeichen nicht ersetzen.

Eingabe Die im Programm eingelesenen Daten.

Eingabe/Ausgabe (E/A) Ein Begriff, der sich auf die Vorgänge und Geräte bezieht, die mit dem Einlesen (Eingabe) der Daten in den Speicher und dem Schreiben (Ausgabe) der Daten aus dem Speicher zusammenhängen.

Eingabeaufforderung Eine Anfrage des Computers, die den Benutzer auffordert, weitere Informationen einzugeben oder eine Aktion durchzuführen.

Emulator Eine mathematische Gleitkomma-Funktionseinheit, die die Eigenschaften eines mathematischen Hardware Koprozessors nachbildet, um dieselben mathematischen Operationen durchführen zu können.

Ereignis Eine Aktion, dessen Verfolgung programmierbar ist. Ereignisse erfassen z.B. das Klicken der Maustaste, die Auswahl von Menüpunkten, die Betätigung der Tastatur oder Zeitabläufe.

Ereignisverfolgung Die Entdeckung von Ereignissen und die Durchführung geeigneter Aktionen, wie zum Beispiel die Verzweigung zu einer entsprechenden Routine.

Erweiterung Der nach dem Punkt stehende Bestandteil eines Dateinamens. *.dat* ist zum Beispiel die Erweiterung des Dateinamens *test.dat*.

Fenster Ein Bildschirmbereich, der eine Datei anzeigt oder in dem Anweisungen eingegeben werden. Der Begriff bezieht sich ausschließlich auf den Bereich des Bildschirms, nicht auch auf den Inhalt der Anzeige. Dieser Begriff bezeichnet ebenfalls einen Ausschnitt eines logischen Koordinatensystems, das mit Hilfe der Graphikanweisung **WINDOW** auf die physikalischen Koordinaten abgebildet wird. Siehe "Direktfenster", "Arbeitsfenster" und "Debug-Fenster".

FLAG-Register Ein Register, das Informationen über den Status der Zentraleinheit und die letzte ausgeführte Operation enthält.

Formale Parameter Variablen, denen die an eine Funktion oder an ein Unterprogramm übergebenen Argumente zugewiesen werden (siehe "Parameter").

FUNCTION Ein Block von Anweisungen, der mit **FUNCTION** beginnt, mit **END FUNCTION** endet und eine Prozedur definiert. **FUNCTION**-Prozeduren können anstelle der in älteren BASIC-Versionen verwendeten **DEF FN**-Funktionen benutzt werden.

Funktionsdefinition Ein Anweisungsblock, der den Namen einer Funktion, deren formale Parameter und die funktionsdefinierenden Vereinbarungen und Anweisungen angibt (bzw. die Funktion ausführt).

Funktionsvereinbarung Eine Vereinbarung, die den Namen und den Typ der Funktion bestimmt, ohne die Funktion selbst explizit zu definieren.

Ganzzahl Eine ganze Zahl, die im Computer als eine 16-Bit-Binärzahl im Zweierkomplement dargestellt wird. Ganzzahlen liegen im Bereich von -32.768 bis +32.767. Siehe "Lange Ganzzahl".

Geltungsbereich Der Bereich von Anweisungen, über den eine Variable oder Konstante definiert wird. Siehe "Globale Konstante", "Globale Variable", "Lokale Konstante" und "Lokale Variable".

Gerätename Der einem Gerät zugeordnete Name, wie PRN, LPT1, LPT2 oder LPT3 für einen Drucker, der von DOS wie eine Datei behandelt wird.

Gleitkomma Eine der wissenschaftlichen Schreibweise ähnliche Schreibweise zur geeigneten Darstellung von sehr großen und sehr kleinen Zahlen sowie von Bruchzahlen. Eine Gleitkommazahl besteht aus zwei Teilen, dem Exponenten und der Mantisse (siehe "IEEE-Format").

Globale Konstante Im Modul-Ebenen-Code definierte symbolische Konstante, die im gesamten Programm verfügbar ist.

Globale Variable Eine Variable, die im gesamten Programm verfügbar ist und mit den Anweisungen **COMMON**, **DIM** oder **REDIM** in Verbindung mit **SHARED** deklariert wird.

Globales Symbol Ein Symbol, das im gesamten Programm verfügbar ist. Funktionsnamen stellen stets globale Symbole dar. Siehe auch "Symbol" und "Lokales Symbol".

Haltepunkt Eine bestimmte Stelle, an der die Programmausführung während der Fehlerbeseitigung anhält. Die Beschreibung der Haltepunkte ist unter "Stoppbedingung" zu finden.

Hauptmodul Das Modul, das die erste ausführbare Anweisung (den Startpunkt des Programms) enthält. Siehe "Modul".

Heap Ein Bereich des Direktzugriffspeichers, der von BASIC verwendet wird, um unter anderem Variablen und Datenfelder zu speichern.

Hervorheben Inverse Darstellung eines Bereichs in einem Textfeld, Fenster oder Menü, der den aktuell gewählten Befehl oder den zum Kopieren oder Löschen ausgewählten Text markiert.

Hexadezimal Das Zahlensystem mit der Basis 16, das die Zeichen 0 bis F (die Buchstaben A bis F stellen die Zahlen 10 bis 15 dar) verwendet. Hexadezimale Zahlen werden oft in der Programmierung benutzt, da sie leicht ins Binärsystem mit der Basis 2, das der Computer selbst verwendet, umzuwandeln sind.

Hyperverbindung Eine Verbindung zwischen zwei zusammenhängenden Informationen, die im Hilfsbildschirm zwischen zwei Dreiecken steht.

IEEE-Format Eine international anerkannte Methode zur Darstellung von Gleitkommazahlen innerhalb der Hardware. Das IEEE-Format erzeugt genauere Ergebnisse als das in früheren QuickBASIC-Versionen verwendete Microsoft-Binär-Format und vereinfacht den Einsatz eines mathematischen Koprozessors. Die Abkürzung IEEE steht für Institute of Electrical and Electronics Engineers, die Organisation, die diesen Standard entwickelt hat. Siehe "Gleitkomma".

Include-Datei Eine Textdatei, die mit dem Metabefehl **$INCLUDE** in einem Programm mitkompiliert wird.

Intelligenter Editor Sammelbegriff für sämtliche fortgeschrittene Einrichtungen von QuickBASIC, einschließlich der Syntaxüberprüfung, der automatischen Großschreibung der Schlüsselwörter und der automatischen Prozedurvereinbarung.

Interpreter Ein Programm, das BASIC-Anweisungen einzeln übersetzt und ausführt. Im Unterschied zum Compiler, der sämtliche Anweisungen vor Ausführung kompiliert, übersetzt ein Interpreter jede einzelne Anweisung vor deren Ausführung. BASICA ist ein Sprach-Interpreter.

Klicken Betätigen einer Maustaste, während der Mauszeiger auf ein Objekt auf dem Bildschirm zeigt.

Kompilieren Das Übersetzen von Anweisungen in eine Form, die vom Computer ausgeführt werden kann.

Konstante Ein Wert, der sich während der Programmausführung nicht ändert.

Kontrollfeld Zwei Klammern in einem QuickBASIC-Dialogfeld, die das Ein- und Ausschalten von Optionen ermöglichen. Eine eingeschaltete Option wird durch ein von den beiden Klammern eingeschlossenes X im Dialogfeld gekennzeichnet.

Koprozessor 8087, 80287 oder 80387 Intel-Hardware, die sehr schnelle Zahlenverarbeitung ermöglicht.

Kurze Adresse Ein Speicherbereich, der nur mit dem am Anfang des Segments stehenden Offset angegeben wird und nicht mehr als zwei Bytes erfordert. Siehe auch "Lange Adresse".

Lange Adresse Eine Speicherzelle, die durch ein Segment (Bereich eines Blocks mit 64K) und einen Offset vom Anfang des Segments angegeben wird. Die auch als Intersegmentadressen bezeichneten langen Adressen erfordern vier Bytes - zwei für das Segment und zwei für den Offset.

Lange Ganzzahl Eine ganze Zahl, die innerhalb der Maschine als eine 32-Bit-Binärzahl im Zweierkomplement dargestellt wird. Lange Ganzzahlen liegen im Bereich von -2.147.483.648 bis +2.147.483.647. Siehe "Ganzzahl".

Laufzeit Die Zeit, während der ein Programm ausgeführt wird. Die Laufzeit bezieht sich auf die Ausführungszeit eines Programms und nicht auf die Ausführungszeit des Compilers oder des Linkers. Einige als Laufzeitfehler bezeichnete Fehler können nur während der Programmausführung entdeckt werden.

Laufzeit-Bibliothek Eine Datei, die die zur Implementierung bestimmter Sprachfunktionen erforderlichen Routinen enthält.

Laufzeit-Modul Ein Modul, das die meisten zur Implementierung einer Programmiersprache notwendigen Routinen enthält. In QuickBASIC ist das Laufzeit-Modul eine Datei mit dem Namen *brun45.exe*, die größtenteils aus einer Bibliothek von Routinen besteht.

Logische Koordinaten Zahlenpaare (zum Beispiel *x* und *y*), die einen beliebigen Punkt eines zweidimensionalen Raumes eindeutig definieren. Die Zahlen *x* und *y* können jede reelle Zahl sein. Logische Koordinaten können mit der Graphikanweisung **WINDOW** auf dem Bildschirm abgebildet werden. Sie ermöglichen das Zeichnen von Daten mit Koordinaten, die der Graphik oder dem Bild angepaßt sind, ohne die aktuellen Bildschirmkoordinaten zu berücksichtigen. Siehe "Physikalische Koordinaten".

Logisches Gerät Ein symbolischer Name, mit dem der Benutzer ein physikalisches (reelles) Gerät nachbilden kann.

Lokale Konstante Eine Konstante, deren Geltungsbereich auf eine bestimmte Code-Einheit, wie den Modul-Ebenen-Code, oder auf eine Prozedur eines Moduls begrenzt ist. Alle innerhalb einer Prozedur in **CONST**-Anweisungen deklarierten Konstanten sind lokale Konstanten. Siehe "Globale Konstanten".

Lokale Variable Eine Variable, deren Geltungsbereich auf eine bestimmte Code-Einheit, wie den Modul-Ebenen-Code, oder auf eine Prozedur eines Moduls begrenzt ist. In einer Prozedur sind alle Variablen lokal, es sei denn, sie erscheinen in einer **SHARED**-Anweisung innerhalb der Prozedur oder in einer **COMMON SHARED**-, **DIM SHARED**- oder **REDIM SHARED**-Anweisung auf der Ebene des Moduls, das die Prozedurdefinition enthält. Siehe "Globale Konstante".

Lokales Symbol Ein Symbol, das nur innerhalb einer bestimmten Funktion von Bedeutung ist. Ein/e innerhalb einer Funktion als statisch deklariertes Argument oder Variable kann ein lokales Symbol darstellen. Siehe auch "Globales Symbol".

LPT1, LPT2, LPT3 Abkürzungen für die Zeilendrucker, die sich auf die drei Anschlüsse beziehen, an die parallele Drucker angeschlossen werden können.

Marke Eine zur Identifizierung dienende Zeichenkette, die vor einer bestimmten Programmstelle steht oder diese markiert oder bezeichnet. Marken werden anhand von **GOTO**-Anweisungen verwendet.

Mathematischer Koprozessor Eine optionale Hardwarekomponente, wie zum Beispiel ein 8087-, 80287- oder 80387-Chip, die die Geschwindigkeit und Genauigkeit von Berechnungen mit Gleitkommazahlen erhöht. QuickBASIC greift automatisch auf einen mathematischen Koprozessor zurück, falls dieser vorhanden ist, oder emuliert ihn mit Hilfe der Software, falls dies nicht der Fall ist. Siehe "Gleitkomma".

Maus Ein Zeigegerät, das auf einer glatten Unterlage bewegt wird. Die Bewegung des Mauszeigers auf dem Bildschirm entspricht der Bewegung der Maus.

Mauscursor Das invers dargestellte Rechteck, das die aktuelle Position der Maus anzeigt. Der Mauszeiger erscheint nur, wenn eine Maus verfügbar ist.

MCGA (Multicolor Graphics Array) Eine mit einer gedruckten Schaltung versehene Karte, die die Anzeige steuert. Die in der Zentraleinheit eines Computers eingesetzte MCGA-Karte zeigt sowohl Text als auch Graphik in niedriger bis mittlerer Auflösung in höchstens 256 Farben an.

MDA (Monochrome Display Adapter) Eine mit einer gedruckten Schaltung versehene Karte, die die Anzeige steuert. Der MDA zeigt den Text nur einfarbig in mittlerer Auflösung an.

Mehrdimensionales Datenfeld Ein Datenfeld, das mehr als einen Index enthält, zum Beispiel `x(5, 35)`. Ein mehrdimensionales Datenfeld wird auch als aus mehreren Datenfeldern bestehendes Datenfeld bezeichnet.

Menüzeile Die Zeile im oberen Teil des QuickBASIC-Bildschirms, die die Menübezeichnungen enthält.

Metabefehle In Kommentaren einer Quelldatei eingeschlossene Sonderbefehle, die den Compiler anweisen, während der Kompilierung bestimmte Aktionen durchzuführen. Der Metabefehl **$INCLUDE** weist beispielsweise den Compiler an, Anweisungen aus einer gegebenen Datei zu übersetzen. Siehe "Include-Datei".

Microsoft-Binär-Format Eine Methode zur internen Darstellung von Gleitkommazahlen, die von früheren QuickBASIC-Versionen (vor Version 3.0) verwendet wurde. Siehe "Gleitkomma" und "IEEE-Format".

Modul Ein getrennt gespeicherter Abschnitt des Programmcodes, der auch separat kompiliert werden kann. Jedes Programm besteht aus mindestens einem Modul (dem Hauptmodul).

Modul-Ebenen-Code Programmanweisungen innerhalb eines Moduls, die außerhalb einer **SUB**- oder **FUNCTION**-Definition stehen. Fehler- oder Ereignisbehandlungscode sowie deklarierende Anweisungen, wie zum Beispiel **COMMON**, **DECLARE** und **TYPE**, dürfen nur auf Modul-Ebene erscheinen.

Modulare Programmierung Eine Programmiertechnik, die das Programm in funktionelle Blöcke aufteilt, die jeweils eine bestimmte Aufgabe übernehmen.

Nullzeichen Das als 0-Wert codierte ASCII-Zeichen.

Nur zu lesende Datei Eine Datei, deren Attribute so festgelegt wurden, daß sie zwar gelesen, jedoch nicht geändert werden kann.

Objektdatei Eine mit der Erweiterung *.obj* versehene Datei, die verschiebbaren Maschinencode enthält, der beim Kompilieren eines Programms erzeugt und vom Linker zur Erstellung einer ausführbaren Datei verwendet wird.

Offset Die Byteanzahl vom Anfang eines Segments bis zu einem bestimmten Byte in diesem Segment.

Operand Der Wert einer Konstanten oder Variablen, mit dem Operationen durchgeführt werden.

Operator Ein oder mehrere Symbole, die die mit den Operanden eines Ausdruckes durchgeführten Operationen angeben.

Overlay Bestandteil eines Programms, der nur bei Bedarf aus der Diskette im Arbeitsspeicher eingelesen wird.

Palette Die für einen bestimmten Anzeigemodus verfügbaren Farben. Der CGA-Modus funktioniert mit einer vorgegebenen Farbenpalette. Die Farbmodi EGA, MCGA und VGA arbeiten mit einer umdefinierbaren Farbenpalette.

Parallele Schnittstelle Der Anschluß, an dem üblicherweise der Drucker angeschlossen wird.

Parameter Ein Variablensymbol in einer **DEF FN**, **FUNCTION** oder **SUB**-Anweisung. Beim Aufruf der Funktion oder des Unterprogramms werden die Parameter durch tatsächliche Variablen und Werte ersetzt.

Pfadname Die vollständige Angabe des Speicherplatzes einer Datei. Normalerweise umfaßt ein Pfadname den Buchstaben des Laufwerks, einen Doppelpunkt, einen oder mehrere Verzeichnisnamen und den jeweiligen Dateinamen.

Physikalische Koordinaten Die Bildschirmkoordinaten. Die physikalischen Koordinaten (0, 0) beziehen sich auf die obere linke Ecke des Bildschirms, wenn vorher kein graphisches Darstellungsfeld anhand einer **VIEW**-Anweisung definiert wurde. In diesem Fall sind (0, 0) die Koordinaten der linken oberen Ecke des Darstellungsfeldes. Der Bereich der physikalischen Koordinaten hängt vom Bildschirmadapter, dem Monitor, sowie von den Angaben der Anweisung **SCREEN** ab. Physikalische Koordinaten werden in Bildpunkten gemessen. Siehe auch "Absolute Koordinaten" und "Logische Koordinaten".

Platzhalter Ein DOS-Zeichen (? oder *), das in den Dateinamen-Verweisen zu einem oder mehreren Zeichen erweitert werden kann.

Potenzierung Eine Zahl *x* mit *y* potenzieren, wobei die Zahl *x* *y*-mal mit sich selbst multipliziert wird. 2^3 bedeutet zum Beispiel `2 * 2 * 2` oder 2 hoch 3.

Programm Eine oder mehrere gebundene Module, die einen vollständigen Satz von ausführbaren Anleitungen bilden.

Programmschritt Hervorheben und Ausführen der nächsten Anweisung.

Prozedur Ein allgemeiner Begriff für eine **SUB** oder eine **FUNCTION**. Siehe "**FUNCTION**" und "**SUB**".

Prozeduraufruf Der Aufruf einer **SUB** oder **FUNCTION**.

Punkte zur Textmarkierung Eine Stelle im Programmcode, die durch Betätigen eines Kontrollcodes wiedergefunden werden kann.

QB-Ratgeber Die ausführliche, kontextabhängige On-Line-Hilfe zur BASIC-Sprache.

Quelldatei Eine Textdatei, die BASIC-Anweisungen enthält. Obwohl alle Module Quelldateien darstellen, sind nicht alle Quelldateien Module (z.B. Include-Dateien).

QuickBASIC verwendet Quelldateien mit zwei unterschiedlichen Formaten. Im QuickBASIC-Format gespeicherte Quelldateien werden schnell geladen, können jedoch nicht direkt mit einem normalen Texteditor gelesen oder manipuliert werden. Textdateien (oder ASCII-Dateien) können mit einem beliebigen Texteditor gelesen und bearbeitet werden.

Quick-Bibliothek Eine Datei, die mit dem Befehl **Bibliothek erstellen** oder mit der Option /Q des Befehls **LINK** angelegt wurde. Eine Quick-Bibliothek enthält Prozeduren, die während der Programmentwicklung in der QuickBASIC-Umgebung verwendet werden können. In früheren QuickBASIC-Versionen wurden Quick-Bibliotheken als Benutzerbibliotheken bezeichnet. Benutzerbibliotheken können jedoch nicht als Quick-Bibliotheken eingesetzt werden, und umgekehrt.

Referenzaufruf Siehe "Übergabe als Referenz".

Referenzzeile Die im unteren Teil des Umgebungsbildschirms vorhandene Zeile, die häufig verwendete Befehle anzeigt.

Register Bereiche des Prozessors, in denen Daten während der Verarbeitung auf Maschinenebene temporär gespeichert werden. Die in der 8086-Prozessorfamilie verwendeten Register sind: AX, BX, CX, DX, SP, BP, SI, DI, DS, ES, SS, CS, IP und das Register FLAG. Siehe "FLAG-Register".

Rekursion Der Vorgang, durch den sich ein Unterprogramm selbst aufruft. Die Rekursion ermöglicht die schnelle und effiziente Codierung einiger Algorithmen.

Relationale Datenbank Eine Datenbank, in der ein beliebiges Feld oder ein Datensatz mit jedem anderen Feld oder Datensatz verbunden werden kann.

Richtungstasten Ein Tastenblock, der in zahlreichen Computer-Tastaturen vorhanden ist und den Cursor nach links, nach rechts, nach oben oder nach unten bewegt. Diese Tasten sind üblicherweise mit Pfeilen gekennzeichnet.

Rollen Den Text nach oben bzw. nach unten oder nach links bzw. nach rechts bewegen, um Textteile zu sehen, die nicht mehr auf den Bildschirm passen.

Schlüsselwort Der Name einer BASIC-Anweisung, -Funktion, oder eines BASIC-Operators. **MOD**, **OPEN**, **OR**, **PRINT** und **SIN** sind Beispiele von Schlüsselwörtern.

Schnittstelle Eine gemeinsam benutzte Verbindungsstelle zweier Systeme oder Einheiten, wie Laufwerk und Computer, Benutzer und Programm.

Segment Ein Speicherbereich, der höchstens 64K beträgt und Code oder Daten enthält.

Seitenverhältnis Das Verhältnis Breite zu Höhe eines Bildschirms. Dieser Begriff wird oft zur Darstellung der Bildpunkte benutzt. Wird ein Bildpunkt als Rechteck betrachtet, besteht sein Seitenverhältnis aus dem Verhältnis Breite zu Höhe. Die typische IBM PC-Anzeige weist ein Seitenverhältnis von ungefähr 4:3 auf.

Selbständige Datei Eine ausführbare Datei, die ohne eine Laufzeitdatei gestartet werden kann. Dazu muß das kompilierte Programm mit *bcom45.lib* gebunden werden.

Sequentielle Datei Eine Datei, die nur der Reihe nach vom ersten bis zum letzten Byte gelesen und geschrieben werden kann. Siehe "Direktzugriffsdatei".

Serielle Schnittstelle Der Datenübertragungsanschluß (COM1, COM2, COM3 oder COM4), an dem Geräte, wie ein Modem oder ein serieller Drucker, angeschlossen werden können.

Standard Eine Einstellung, die dann verwendet wird, wenn keine andere angegeben ist.

Standard-Ausgabe Das Gerät, an das ein Programm die Ausgabe sendet, falls diese nicht umgelenkt wurde. In DOS erfolgt die Standard-Ausgabe über den Bildschirm.

Standard-Datentyp Der von BASIC für Daten verwendete Typ, wenn kein anderer Typ mit einer **DEF***typ*-Anweisung angegeben wird. Der Standardtyp für numerische Daten ist einfache Genauigkeit.

Standard-Eingabe Das Gerät, aus dem ein Programm die Eingabe einliest, falls die Eingabe nicht umgelenkt wurde. In DOS erfolgt die Standard-Eingabe über die Tastatur.

Stapel Ein Bereich des Direktzugriffsspeichers, der von BASIC zum Speichern von Daten wie Zwischenergebnisse arithmetischer Berechnungen, automatische Variablen in Prozeduren sowie Prozeduraufrufe verwendet wird.

Statisches Datenfeld Ein Datenfeld, dem ein Speicherplatz während der Programmkompilierung zugewiesen wird. Der Speicherplatz eines statischen Datenfeldes ändert sich nicht während der Programmausführung.

Stoppbit Ein in der seriellen Datenübertragung benutztes Signal, das das Ende eines Zeichens markiert.

Stoppbedingung Ein Ausdruck (wie `X = 3`), der die Programmausführung anhält, sobald er wahr wird. Stoppbedingungen verlangsamen die Ausführung. Siehe "Haltepunkt".

SUB Eine Code-Einheit, die von den Anweisungen **SUB** und **END SUB** begrenzt wird. Eine **SUB** (manchmal als **SUB**-Prozedur bezeichnet) bietet eine Alternative zu der älteren BASIC-Unterroutine **GOSUB**.

Symbol Ein Name, der einen Speicherplatz identifiziert.

Symbolische Konstante Eine Konstante, die von einem Symbol und nicht der Konstanten selbst dargestellt und mit Hilfe der Anweisung **CONST** definiert wird.

Text Normale, ausgebbare Zeichen, die die Groß- und Kleinbuchstaben des Alphabets, die Ziffern 0 bis 9 und die Satzzeichen umfassen.

Textdatei Eine Datei, die nur Text enthält.

Texteditor Ein Programm zum Erstellen und Ändern von Textdateien.

Textfeld Ein Feld, in dem zum Ausführen von Befehlen erforderliche Informationen eingegeben werden und das innerhalb eines Dialogfeldes erscheint.

Titelleiste Eine Leiste im oberen Teil des Arbeitsfensters, die den Namen der aktuell in diesem Fenster angezeigten Textdatei oder des Moduls (und ggf. der Prozedur) angibt.

Typ Das numerische Format einer Variablen. Typen umfassen Ganzzahlen, lange Ganzzahlen, Zahlen einfacher Genauigkeit und doppelter Genauigkeit und können auch benutzerdefiniert sein.

Typenkontrolle Eine Operation des Compilers, die die Gültigkeit von Operanden und Operatoren sowie die Übereinstimmung der Argumente eines Funktionsaufrufes mit den entsprechenden formalen Parametern der Funktionsdefinition überprüft.

Typenvereinbarung Eine Vereinbarung, die Variablennamen als Bestandteil einer bestimmten Datenstruktur oder eines bestimmten Typs definiert.

Übergabe als Referenz Übergabe der Adresse eines Arguments an eine **SUB** oder **FUNCTION**. Dieser Vorgang erlaubt einer Prozedur, die ihr übergebenen Argumentenwerte zu verändern.

Übergabe als Wert Übergabe des Wertes (nicht der Adresse) eines Arguments an eine **SUB** oder **FUNCTION**.

Überlauf Ein Fehler, der auftritt, wenn der einer numerischen Variablen zugewiesene Wert außerhalb des für diesen Variablentyp zulässigen Bereichs liegt.

Umlenken Die Angabe eines Geräts (das nicht das Standard-Gerät darstellt), aus dem das Programm Eingabedaten erhalten oder an dem es Ausgabedaten senden wird. Standardmäßig erfolgt die Eingabe über die Tastatur und die Ausgabe über den Bildschirm.

Umschalter Eine Funktionstaste oder Menüwahl, die eine Eigenschaft ausschaltet, wenn diese eingeschaltet ist, oder diese Eigenschaft einschaltet, wenn sie ausgeschaltet ist. Umschalten bezeichnet das Umkehren des Zustands einer Einrichtung. Der Befehl **Verfolgen ein** aus dem Menü **Debug** schaltet beispielsweise die Ablaufverfolgung ein und aus.

Ungelöster Verweis Eine in einem Modul deklarierte Prozedur, die jedoch in einem anderen Modul, das mit dem ersten Modul gebunden ist, nicht definiert wird. Ungelöste Verweise werden normalerweise durch Schreibfehler oder durch Auslassen des Modulnamens verursacht, der die Prozedur aus der LINK-Befehlszeile enthält.

Unterlauf Ein Fehler, der vom Ergebnis einer Berechnung verursacht wird, das für die interne Behandlung im Computer zu klein ist.

Unterprogramm Siehe "SUB".

Unterroutine Eine Code-Einheit von BASIC, die von der Anweisung **RETURN** beendet wird. Die Programmkontrolle wird mit einer **GOSUB**-Anweisung an eine Unterroutine übergeben, die auch als **GOSUB**-Block bezeichnet wird.

Variable Ein Wert, der sich normalerweise während der Programmausführung ändert.

Vereinbarung Eine Konstruktion, die Namen und Eigenschaften von Variablen, Funktionen oder Typen miteinander verknüpft.

Verkacheln Das Ausfüllen einer Figur mit einem Muster anstelle einer durchgängigen Farbe mit Hilfe der Anweisung **PAINT**.

Verzeichnis Ein Dateiindex, der von DOS zum Auffinden von Dateien auf der Diskette benutzt wird.

Verzeichnisfeld Ein Feld, das eine Reihe von Elementen aufführt. Das Feld, das die Dateien im Dialogfeld **Programm laden** auflistet, ist beispielsweise ein Verzeichnisfeld. Siehe auch "Dialogfeld".

VGA (Video Graphics Array) Eine mit einer gedruckten Schaltung versehene Karte, die die Anzeige steuert. Ein VGA zeigt sowohl Text als auch Graphiken in mittlerer bis hoher Auflösung und bis zu 256 Farben an.

Vorrang Die relative Position eines Operators in der Hierarchie, die die Reihenfolge bestimmt, in der die Ausdrücke ausgewertet werden.

Wertaufruf Siehe "Übergabe als Wert".

Wurzelverzeichnis Das von DOS auf jeder Diskette erstellte Hauptverzeichnis, das sich im hierarchischen Dateisystem an oberster Stelle befindet.

Zeichenkette Eine Zeichenfolge, die oft durch einen symbolischen Namen identifiziert wird. Im nachfolgenden Beispiel stellt `"Hallo"` eine Zeichenkette dar: `PRINT "Hallo"`. Eine Zeichenkette kann eine Konstante oder Variable sein.

Zeiger Siehe "Mauszeiger".

Ziehen Den Mauscursor zu einem Objekt auf dem Bildschirm bringen, danach die Maus bei gedrückter Maustaste bewegen. Das Objekt bewegt sich in die gleiche Richtung wie die Maus. Wenn das Objekt die gewünschte Stelle erreicht, Maustaste loslassen.

Zuweisung Bezeichnet den Vorgang, durch den ein Variablen ein Wert zugeordnet wird. Die Anweisung `X = 3` weist *x* den Wert 3 zu.

Zweierkomplement Die Zahlendarstellung mit der Basis 2, mit der sowohl negative als auch positive Werte dargestellt werden. Die negativen Werte werden durch Komplementieren der Bits und Addieren der Zahl 1 zu dem Ergebnis gebildet.

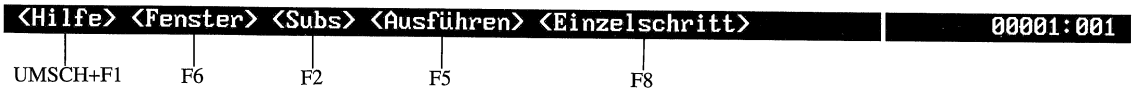
Zweimal klicken Zweifache Betätigung der Maustaste in rascher Folge, während der Mauscursor auf ein Objekt auf dem Bildschirm zeigt.

Zwischenspeicher Ein Speicherbereich, der einen kopierten oder ausgeschnittenen Textabschnitt enthält. Der Zwischenspeicher kann nicht mehrere Textabschnitte enthalten der zuletzt eingefügte Text ersetzt den bereits vorhandenen Text. Der Text im Zwischenspeicher kann mit Hilfe des Befehls **Einfügen** in eine Datei kopiert werden.

Anhang
Referenzzeilen: Tastenkurzkombinationen für die Befehlsflächen

Falls Sie keine Maus besitzen, können Sie die Befehle der vier verschiedenen Referenzzeilen (Bearbeiten, Ausführen und Debuggen, Direktfenster, Hilfe) mit den in diesem Anhang beschriebenen Tastenkombinationen zur direkten Befehlseingabe ausführen. Wenn Sie eine Maus benutzen, genügt ein einfaches Anklicken der Befehlsflächen zur Ausführung der Befehle.

Bearbeiten

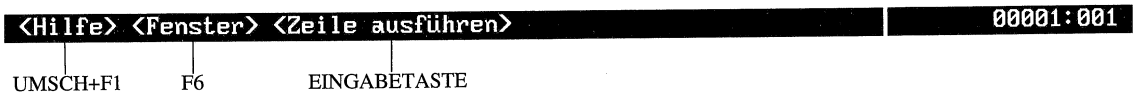


Tasten-kombination	*Funktion*
UMSCH+F1	ruft **Überblick** aus dem Menü **Hilfe** auf.
F6	wechselt das aktive Fenster.
F2	entspricht dem Befehl **SUBs** aus dem Menü **Ansicht**.
F5	entspricht dem Befehl **Start** aus dem Menü **Ausführen**.
F8	führt das Programm im Einzelschrittmodus aus.

Ausführen und Debuggen

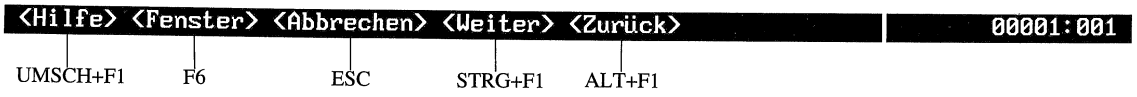
Tasten-kombination	*Funktion*
UMSCH+F1	ruft **Überblick** aus dem Menü **Hilfe** auf.
F5	setzt das Programm mit der Anweisung fort, bei der das Programm angehalten wurde.
F9	legt in der Zeile, in der sich der Cursor befindet, einen Haltepunkt fest. Ist bereits ein Haltepunkt festgelegt, wird er durch Betätigung von F9 gelöscht.
F8	führt das Programm im Einzelschrittmodus aus.

Direktfenster



Tasten-kombination	*Funktion*
UMSCH+F1	ruft **Überblick** aus dem Menü **Hilfe** auf.
F6	wechselt das aktive Fenster.
EINGABETASTE	führt die aktuelle Zeile des Direktfensters aus.

Hilfe



Tastenkombination	*Funktion*
UMSCH+F1	ruft **Überblick** aus dem Menü **Hilfe** auf.
F6	wechselt das aktive Fenster.
ESC	löscht den Hilfsbildschirm und kehrt zum zuletzt aktiven Fenster zurück.
STRG+F1	ruft den nächsten Hilfsbildschirm auf.
ALT+F1	geht einen Hilfsbildschirm zurück. (QuickBASIC speichert die 20 zuletzt aufgerufenen Hilfsbildschirme.)

Index

•(Punkt) neben Befehl 2.8
...(Ellipsenpunkte) mit Menüpunkt 2.9
80287 G.8
80387 G.8
8087 G.8

A

Abbrechen, Befehle 2.4
Absolute Koordinaten G.1
Adresse
  Beschreibung G.1
  kurze G.9
Aktives Fenster
  Beschreibung 3.1
  Größe ändern 3.8
  umschalten 10.17
  wechseln 10.17
  Zeilen- und Spaltenzähler 10.6
Aktivierte Ereignisse G.1
Aktuelle Anweisung, Option (Bildschirm, Befehl) 2.13
Aktuelle Laufwerke G.1
Aktuellen Wert anzeigen
  Dialogfeld, Abbildung 18.5
  Symbolhilfe 7.6
Aktuelles Verzeichnis G.1
Algorithmus G.1
Alle Haltepunkte löschen, Befehl 9.2
Alles speichern, Befehl 11.12
ALT-Taste
  ASCII-Zeichen höherer Ordnung eingeben 12.8
  Beschreibung 10.8
  Dialogfelder, verwenden in 2.11
  Menüs und Befehle 2.4
AND-Operator 4.20
Ändern
  aktives Fenster 10.17
  Bildschirmanzeige 2.11
  Fenstergröße 3.8, 10.18, 14.6
  Hauptmodule 16.15
  Programme 10.24
  Standard-Suchpfade 20.3
  Text 15.6
  Variablenwerte 3.11
Ändern der Standard-Suchpfade 20.3
Ändern, Befehl
  Beschreibung 15.6
  Dialogfeld, Abbildung 6.14, 15.6
Animation G.1
Anpassen der Bildschirmanzeige 2.11, 20.2
Ansicht, Menü
  Anzeigen Include-Datei, Befehl 14.11
  Ausgabebildschirm, Befehl 14.8
  Bearbeiten Include-Datei, Befehl 14.9
  Befehle (Liste) 14.1
  Nächste Anweisung, Befehl 14.7
  Nächste SUB, Befehl 14.5
  SUB's, Befehl 10.16, 14.1
  Teilen, Befehl 10.16, 14.6
Ansichtsprozeduren 14.3
Anweisung
  Beginn der Ausführung von 18.17
  ERROR 10.25
Anweisung überprüfen *Siehe* Syntaxüberprüfung
Anweisungen
  CHAIN 16.9
  COMMON 16.9
  DECLARE 13.10
Anweisungen (*Fortsetzung*)
  Definition G.2
  DIM 16.10
  DO, Hilfe, Abbildung 8.2
  GOSUB 16.10
  IF...THEN...ELSE, Hilfe, Abbildung 7.2
  INPUT 4.15, 16.10
  nächste auszuführende 14.7
  PRINT 3.7
  RETURN 16.10
  SELECT CASE
    Beschreibung 8.8
    Hilfsanzeige, Abbildung 8.5
  STEP 4.24
  zusammenstellen anhand der Hilfe 7.1
Anzeigbar G.2
Anzeigen Include-Datei, Befehl 14.11
Anzeigevariable löschen, Befehl
  Debug-Fenster schließen 9.12
  Variablen löschen 9.2
Anzeigevariablen 17.5
Anzeigevariablen, debuggen 17.5
Apostroph (') xx
Arbeitsfenster
  Abbildung 2.3, 10.6
  Beschreibung 3.2-3, 10.16
  Definition G.2
  löschen 3.6
  Programm laden in 3.3
  programmieren in 3.6-7
  Prozeduren in 9.4
  Prozeduren verschieben 5.9
  teilen 10.16
Argumente G.2

Arithmetische
  Gleitkomma 16.11
  Überlauf 16.10
  Umwandlung G.2
ASCII-Zeichen
  Entsprechungen zu Steuerfolgen 12.8
  höherer Ordnung eingeben 12.8
  Satz G.2
ASCII-Zeichen höherer Ordnung 12.8
Attribute G.2
Aufruf als Referenz *Siehe* Übergabe, als Referenz
Aufruf als Wert *Siehe* Übergabe, als Wert
Aufrufe, Menü 19.1
Ausdrücke
  Anwendung 4.12
  Anzeigevariablen 17.5
  Definition 4.12, G.2
  einfache 4.12
Ausführbare Dateien
  anlegen 16.6-8
  Definition G.2
  Quick-Bibliotheken 16.8
  selbständige 9.13, G.13
Ausführen bis zum Cursor, Befehl 17.7
Ausführen, Menü
  Bibliothek erstellen, Befehl 16.13
  Debug, Befehle 9.2
  EXE-Datei erstellen, Befehl 16.7
  Hauptmodul bestimmen, Befehl 16.15
Ausführen, Option (QB) 10.2
Ausführung
  alle Anweisungen der Prozeduren 9.3
  Programme
    bis zum Cursor 7.7, 9.3
    rechte Maustaste 20.5
    unterbrechen 7.4
  Steuerungsbefehle *Siehe* Auswählen, Befehle
Ausgeben
  ASCII-Steuerfolgen niedriger Ordnung 12.9
  ASCII-Zeichen höherer Ordnung 12.8
Ausschneiden, Befehl 6.9, 9.4, 13.3
Auswählen
  Befehle
    EINGABETASTE 10.9
    Kurzmenüs 2.7-8
    Maus 10.27
    Menüs 2.4, 2.7-8, 10.8
    Tastatur 10.8
  Definition G.3
  Menüs 10.8
  Programme 3.5
  Text
    Bearbeitungsfunktionen 12.2
    Beschreibung 6.9
    normal darstellen 6.9
    Vorsicht 6.9
    Zusammenfassung 12.12
Automatische Formatierung 12.8
Automatische Syntaxüberprüfung 12.4, 20.6

B

/B-Option (QB) 10.2
BASIC-Befehlszeilen-Compiler *Siehe* BC
BASICA-Programme, ausführen mit QuickBASIC 16.2
Basisname 16.13, G.3
BC 16.6
BCOM45.LIB 16.6-7
Bearbeiten Include-Datei, Befehl 14.9
Bearbeiten
  Befehle 12.9
  Include-Dateien 14.9
  Material aus Hilfe 8.8
Bearbeiten, Menü
  Ausschneiden, Befehl 13.3
  Befehle (Liste) 13.1
  Kopieren, Befehl 13.4
  Löschen, Befehl 13.6
  Neue FUNCTION, Befehl 13.11
  Neue SUB, Befehl 13.7
  Rückgängig machen, Befehl 13.2
  Syntaxüberprüfung, Befehl 12.4-5
Beenden, QuickBASIC 3.15, 11.21
Befehle
  abbrechen 2.4
  Aktuellen Wert anzeigen 9.2
  Aktuellen Werte anzeigen 7.6
Befehle (*Fortsetzung*)
  Alle Haltepunkte löschen 9.2
  Alles speichern 11.12
  Ändern 6.14, 15.6
  Anzeigen Include-Zeilen 14.11
  Anzeigevariable löschen 9.2
  Anzeigevariablen löschen 9.12
  Ausführen bis Cursor 17.7
  Ausgabebildschirm 3.7, 14.8
  Ausschneiden 6.9, 9.4, 13.3
  auswählen
    EINGABETASTE 10.9
    Kurzmenüs 2.7-8
    Maus 10.27
    Menüs 2.4, 2.7-8, 10.8
    Tastatur 10.8
  Bearbeiten Include-Datei 14.9
  Bearbeiten 12.9
  Begriff 2.7, 21.3
  Bildschirm 20.2
  Datei entfernen 11.17
  Datei erstellen 11.13
  Datei laden 11.15
  Debug
    Beschreibung 17.8
    F10 9.3
    F7 9.3, 17.7
    F8 9.3, 9.8
    F9 9.8
    Funktionstasten 9.1
    Menü Debug 9.2
    Tabelle 17.7, 18.1
  direkte Tastenkombinationen 10.10
  DOS-Betriebssystem 11.20
  Drucken
    Dialogfeld 7.8
    Hilfe 7.8
  Drucken
    Definition 11.19
  Einfügen 6.9
  Einzelschrittmodus 17.7
  Ende 11.21
  EXE-Datei erstellen
    Dialogfeld 9.13, 16.6-8
  Halt bei Fehler 17.7
  Haltepunkt ein/aus 9.2
  Hauptmodul bestimmen 16.15
  Hilfe 21.3
  Index 2.7, 7.2, 21.1

Befehle (*Fortsetzung*)
Kopieren 6.9, 13.4
Löschen 13.6
Marke 15.8
markierter Text 15.4
Nächste Anweisung festlegen 17.7
Nächste Anweisung 14.7
Nächste SUB 14.5
Neue SUB 13.7
Neustart 9.2
Programm laden 11.4
Prozedurschritt 17.8
QB 10.2
QuickBASIC starten 10.2
Rechte Maustaste 20.5
Rückgängig machen 13.2
Rückverfolgen ein 17.4
Speichern unter 3.15, 11.11
Speichern 11.9
Start 3.7, 9.2
SUB
Definition 10.16, 14.1
Prozeduren verschieben 5.9, 9.5, 13.7
Suchen 6.12, 15.2
Suchpfade festlegen 20.3
Syntaxüberprüfung 12.4, 20.6
Teilen 10.16, 14.6
Themen 2.7, 21.2
Überblick 2.7, 21.5
Variable anzeigen 3.11, 9.2, 17.5
Verfolgen ein 17.4
Verfügbarkeit 6.10
Vollständiges Menü 20.7
Weiter 9.2, 9.6
Weitersuchen 15.5
Befehlsflächen
Abbildung 2.12
Anmerkungen 7.3
Beispiel 7.4
Beschreibung 2.10, 10.15-16
Bibliothek erstellen und beenden 16.8
EXE erstellen und beenden 16.8
EXE erstellen 16.8
Hilfe 7.3
Programm 7.3
Begriff, Befehl 2.7, 21.3
Beispiel, Befehlsfläche 7.4
Beispielcode
Hilfs-Hyperverbindungen 7.4
kopieren aus Hilfe 8.4
Benennungsprozeduren 13.10
Benutzerdefinierte Typen G.3
Beobachten, Ausdrücke im Debug-Fenster 17.3
Berechnungen
Direktfenster 10.22
Gleitkomma 16.11
Bewegen
Prozeduren
Arbeitsfenster 5.9
Hilfsfenster 10.32
SUBs, Befehl 9.5
Text 6.10, 13.3
Bibliothek erstellen, Befehl 16.13
Bibliothek erstellen und beenden, Befehl 16.14
Bibliotheken
erstellen 16.13
Laufzeit G.9
Quick
Beschreibung 16.8
Definition G.12
selbständige, ausführbare Dateien 16.8
Bild nach oben Taste 10.32
Bild nach unten Taste 10.32
Bildlauffeld 10.28
Bildlaufleisten
Abbildung 2.3, 10.5-6
Definition G.3
Einstellung 2.12, 20.2
Option (Bildschirm, Befehl) 2.12
Bildlaufpfeil, Abbildung 10.5
Bildschirm
Abbildung 2.3
Anzeige anpassen 2.11
Daten anzeigen 4.2
Farben setzen 20.2
löschen 10.7
QuickBASIC 10.4
Bildschirm, Befehl 20.2
Binden G.3
BIOS G.3
Bit G.4
Block G.4
Boolesche Ausdrücke
einfache 4.19
Vorrang der Auswertung 4.21
zusammengesetzte 4.20
BRUN45.EXE 16.9
BRUN45.LIB 16.6-8
Byte G.4

C

/C-Option (QB) 10.3
CGA *Siehe* Farbgraphikadapter
CHAIN-Anweisung 16.9
CHR$-Funktion 12.9
/CMD-Option (QB) 10.3
COMMAND$-Funktion 16.5
COMMON-Anweisung 16.9
Compiler G.4
Compiler/Unterschiede in der Umgebung 16.11
Cursor
Abbildung 10.5
bewegen 3.5, 12.10
Definition G.4

D

Datei entfernen, Befehl 11.17
Datei erstellen, Befehl 11.13
Datei laden, Befehl 11.15
Datei, Menü
Abbildung 10.8
Alles speichern, Befehl 11.12
Befehle (Liste) 11.1
Datei entfernen, Befehl 11.17
Datei erstellen, Befehl 11.13
Datei laden, Befehl 11.15
DOS-Betriebssystem, Befehl 11.20
Drucken, Befehl 11.19
Ende, Befehl 11.21
Programm laden, Befehl 11.4-5
Speichern unter, Befehl 11.11
Speichern, Befehl 11.9
Zusammenführen, Befehl 11.7
Dateiangabe G.4
Dateien
.mak 16.16
ausdrucken 11.19

Dateien (*Fortsetzung*)
ausführbare
Definition G.2
erstellen 16.6-8
Fehlerüberprüfung in 16.10
Quick-Bibliotheken 16.8
selbständige, erstellen 9.13
Daten G.4
einfügen in aktives Fenster 11.7
Include
ansehen und bearbeiten 14.9
Definition 11.14, G.8
nur ansehen 14.11
suchen 14.10
verschachteln 14.10
laden 11.15
Objekt G.11
qb.ini 2.15, 20.1
Quell G.12
selbständige G.13
sequentielle G.13
speichern 11.9-10
Standard-Suchpfade ändern 20.4
Text 11.14, G.14
zusammenführen 11.7
Daten G.4
Datenbank 5.3
Datenfeldelemente, angeben 4.17
Datenfelder
Beschreibung G.4
dynamische G.5
mehrdimensionale G.10
statische G.14
Datenfeldgrenzen
überprüfen 16.10
Datenfeldgrenzen
Definition G.4
Datenfeldvariablen 4.17
Datensätze G.4
Datentypen, Standard
ändern 13.10
Definition G.14
für Prozeduren 13.9
Datenübertragungsanschluß G.4
Debug, Menü
Abbildung 18.1
Befehle (Liste) 18.1
Tabelle 9.2
Debug-Code erstellen, Kontrollfeld
16.7, 16.10, 16.14
Debug-Einrichtungen 17.7
Debug-Fenster
Abbildung 10.26
Befehle 17.3
Beschreibung 3.2, 10.16, 17.4-6
Debug-Programme 3.11
Definition G.4
löschen von Variablen aus 9.2
schließen 9.12
Stoppbedingung 17.4
überwachen von Variablenwerten
3.11-12
Variablen einsetzen in 9.2
Debuggen
Anzeigevariablen 17.4
Befehle
F10 9.3
F7 9.3
F8 9.3, 9.8
F9 9.8
Funktionstasten 9.1
Menü Ausführen 9.2
Menü Debug 9.2
beim Schreiben von Code 17.2
Debug-Fenster 10.26
Direktfenster 17.2
Eigenschaften 17.3
Einführung 17.1
Haltepunkte 17.4
Kontext, Abbildung 10.26
Nächste Anweisung, Befehl 14.7
Prozeduren 9.6, 17.1, 17.4, 17.5, 17.8
Weiter, Befehl 16.4
Debugger
Definition G.5
DECLARE-Anweisung,
Prozedurvereinbarungen 13.10
DEF FN-Funktion
Debug-Fenster 17.5
Definition G.5
DEF*typ* G.5
Einführung 13.9
Dialogfelder
Abbildung 10.13-14
Definition 2.9, G.5
Drucken, Befehl 11.19
Elemente auswählen 10.14-15
löschen 10.7
Optionen setzen 12.11-17
Textfeld 10.14
Zusammenführen, Befehl 11.8
DIM-Anweisung 16.10
Dimensionen G.5
direkte Tastenkombinationen
Befehle 10.10-12
Beschreibung 10.8
Einführung 2.14
Tabelle 10.10-12
Direktfenster
Abbildung 2.3, 10.5-6
Anweisungen ausführen, in 3.9
Berechnungen 10.22
Beschreibung 3.2, 10.16, 10.20
Definition G.5
Einschränkungen 3.11
Fehlerbeseitigung 17.2, 17.6
Fenstergröße, ändern 10.18
Kontext 10.26
Laufzeitfehler simulieren 10.25
Laufzeitfehlermeldungen 17.4
Leistungsvermögen 10.20
nicht-unterstützte Anweisungen
10.22
Programme testen 10.22
Prozeduren aufrufen aus 10.23
Variablenwerte ändern 10.24
wechseln, aktives Fenster 10.17
Division 4.8
DO-Anweisung, Hilfe, Abbildung 8.2
DO...LOOP
Beendigungs-Bedingung 4.26
Definition 4.25
UNTIL-Version 4.27
WHILE-Version 4.26
Dokument G.5
Dokumentkonventionen xix
Doppelte Genauigkeit G.5
DOS, zurückkehren zu 11.20
DOS-Betriebssystem, Befehl 11.20
Drucke, Befehl
Dialogfeld 7.8
Hilfe 7.8
Drucken, Befehl
Beschreibung 11.19
Dialogfeld, Abbildung 11.19
Duplizieren von Text *Siehe* Kopieren
Dynamische Datenfelder G.5

E

E/A *Siehe* Eingabe/Ausgabe
Editor
automatische Formatierung 6.2, 12.8
Schreibkonventionen 6.3

Editor (*Fortsetzung*)
Syntaxüberprüfung 12.4
Text G.14
Texteingabe 12.1
Textmarkierung 12.2
EGA *Siehe* Enhanced Graphics Adapter
Einfache Genauigkeit G.6
Einfache Variablen G.6
Einfüge-Modus 12.1, G.6
Einfügen
Befehle, Zusammenfassung 12.10
Text 12.1
Einfügen von Text 6.10, 8.12, 13.5
Einfügen, Befehl 6.10, 13.5
Eingabe
Definition G.6
Sonderzeichen 12.8
Standard G.14
Text 12.1
Eingabe/Ausgabe (E/A) G.6
Eingabeaufforderungen G.6
Einrücken
Programmblöcke 8.4
Text 6.11, 12.2
Einrückungskontrolle 12.2
Einzelheiten, Befehlsfläche 7.3
Einzelschritt-Modus 17.7
Ellipsenpunkte (...) mit Menüpunkt 2.9
Emulator G.6
Ende, Befehl 11.21
Enhanced Graphics Adapter (EGA) G.5
Entfernen *Siehe auch* Löschen
Module 16.12
Text 13.3, 13.6
Ereignisse G.6
Ereignisverfolgung
Definition G.6
Quick-Bibliotheken 16.14
ERROR-Funktion 10.25
Erstellen
ausführbare Dateien 9.13, 16.6-8
Bibliotheken 16.13
FUNCTION-Prozeduren 13.11-12
Hauptmodule 16.14
neue Unterprogramme 13.8
Prozeduren 9.3
EXE benötigt BRUN45.exe, Option 16.7
EXE erstellen und beenden, Befehlsfläche 16.8
EXE erstellen, Befehlsfläche 16.8
EXE-Datei erstellen, Dialogfeld, Abbildung 9.13, 16.7

F

F1-Taste 3.12
Fällen von Entscheidungen 4.17
Falsch, Darstellung 4.20
Farben *Siehe* Bildschirmfarben
Farbgraphikadapter (CGA) G.4
Fehler
Eingabe 12.6
korrigieren 7.1
Laufzeit
Position angeben 16.8
simulieren 10.25
überprüfen 16.10
Zeilennummern 16.10
Syntaxüberprüfung 12.5
während der Ausführung eines Programms 6.6
Fehler vermeiden 17.2-3
Fehler *Siehe* Debuggen; Fehler
Fehlermeldungen
Hilfe 6.6
löschen 10.7
Fehlerüberprüfung, ausführbare Dateien 16.10
Fenster
aktives
Beschreibung 3.1
wechseln 10.17
aktivieren 3.8
Arbeitsfenster
Beschreibung 3.2-3, 10.16
Definition G.2
ein Programm laden 3.3
programmieren in 3.6-7
Prozeduren in 9.4
teilen 10.16
Arbeitsfenster 10.5
bewegen zwischen 3.8
Debug-Fenster
Befehle 17.4
Beschreibung 3.2, 10.16, 10.26, 17.5
Debug-Programme 3.11
Definition G.4
Einsetzen der Variablen in 9.2
schließen 9.12
Stoppbedingungen 17.4
Überwachen der Variablenwerte 3.11-12
Variablen löschen aus 9.2
Definition G.6
Fenster (*Fortsetzung*)
Direktfenster
Abbildung 10.5-6
ausführen von Anweisungen in 3.9
Berechnungen 10.22
Beschreibung 3.2, 10.16, 10.20
debuggen in 17.6
Definition G.5
Fenstergröße ändern 10.18
Laufzeitfehler simulieren 10.25
nicht unterstützte Anweisungen 10.22
Programme testen 10.22
Prozeduren aufrufen aus 10.23
Umfang 10.21
Variablenwerte ändern 10.24
Größe ändern 3.8, 10.18, 14.6
Hilfe
Abbildung 7.5
Beschreibung 3.2, 3.12, 10.16
bewegen innerhalb 10.32
mit Kurzmenüs verfügbar 3.1
teilen 14.6
Festlegen
Bildlaufleisten 20.2
Bildschirmattribute, aktuelle ausführbare Zeilen 2.13
Bildschirmfarben 20.3
Haltepunkte 9.2
Tab-Abstände 20.2
Fibonacci-Folge 18.4
Finden
Include-Dateien 14.10
Siehe auch Suchen
Text 15.2
Flags G.6
FOR...NEXT-Schleife 4.23
Formale Parameter G.6
Formate
IEEE G.8
Microsoft Binär G.10
Formatierung, automatische 6.2, 12.8
Fortsetzung unterbrochener Programme 9.6
FUNCTION-Prozeduren
Ansicht 14.3
Bezeichnung 13.11
erstellen 13.11
Standard-Datentyp 13.9
Funktionen, Konvertierungsoption (QB) 10.3
Funktionsdefinitionen G.7

Funktionstasten, Debuggen 9.1
Funktionsvereinbarungen G.7

G

/G-Option (QB) 10.2
Ganzzahlen G.7
ganzzahlige Variablen 4.5
Geräte
  logische G.9
  Namen G.7
Geschwindigkeit der
  Bildschirmaktualisierung (QB) 10.2
Gleitkomma
  arithmetisches, ausführbare Dateien 16.11
  Definition G.7
  Variablen 4.5
Globale Symbole G.7
Globale Variablen G.7
GOSUB-Anweisung 16.10
Groß- und Kleinschreibung
  unterscheiden zwischen 6.2

H

/H-Option (QB) 10.2
Halt bei Fehler, Befehl 17.7
Haltepunkt ein/aus 9.2
Haltepunkte
  Beschreibung 17.4
  Definition G.7
  löschen 9.2
  setzen 9.2, 9.8
Hardware-Anforderungen für
  QuickBASIC xvi
Hauptmodul bestimmen, Befehl 16.15
Hauptmodule 5.10, 6.14-15, G.7
Heap G.8
Hervorheben G.8
Hexadezimal G.8
Hilfe
  allgemeine Befehlsflächen 7.3
  Ausdrucken von Text aus 7.8
  Bearbeiten von Material aus 8.8
  Beispiele 7.3
  Beispielprogramme 21.2
  Beschreibung xviii, 2.5
  Dateien
    Disketten Laufwerk-System 10.33
    Festplatten Laufwerk-System 10.33
Hilfe (*Fortsetzung*)
  fehlende Hilfsdateien 10.33
  Fehlermeldungen 6.6
  Hilfe für Probleme 10.33
  Hyperverbindung 3.13, 10.29
  kontextabhängige 3.12
  Kopieren von Beispielen aus 8.4
  Kopieren von Syntaxblöcken aus 8.1
  Punkte zur Textmarkierung 10.30
  QB-Ratgeber 10.33
  rechte Maustaste 20.5
  Schlüsselwörter
    Abbildung 7.3
    Befehlsflächen 7.3
    Beschreibung 7.2
    Listing mit Index 21.1
  Sprachhilfe 10.30
  Suchen nach Hilfsdateien 10.32-33
  Suchpfade festlegen, Befehl für Dateien 10.33
  Syntax 3.12
  Tasten 10.32
  Überblick 21.5
  Übersicht des Systems 21.2
  Verwendung xv, 2.5, 10.29
  zusammenstellen von Anweisungen mit 7.1
Hilfe auf Diskette 10.33
Hilfe auf Festplatte 10.33
Hilfe, Menü
  Befehle (Liste) 21.1
  Begriff, Befehl 21.3
  Einführung 2.5
  Index, Befehl 21.1
  Themen, Befehl 21.2
  Überblick, Befehl 21.5
Hilfsfenster
  Abbildung 7.5
  Beschreibung 3.2, 3.12, 10.16
  bewegen in 10.32
Hohe Auflösung, Bildschirm-Option (QB) 10.2
Hyperverbindungen 3.13, 10.30-31

I

IEEE-Format G.8
IF...THEN...ELSE-Anweisung
  Definition 4.21
  Hilfe, Abbildung 7.2
  Syntax 4.21
Include-Dateien
  ansehen und bearbeiten 14.9
  Definition 11.14, G.8
  nur ansehen 14.11
  Suchen 14.10
  verschachteln 14.10
$INCLUDE-Metabefehl 14.10, 16.16
Index, Befehl
  Anzeige, Abbildung 21.2
  Beschreibung 2.7, 21.1
  verwenden 7.2
INPUT-Anweisung 4.15, 16.10
Installieren von QuickBASIC 1.1
Intelligenter Editor *Siehe auch* Editor
  automatische Prozedurvereinbarungen 6.8
  Beschreibung 6.1
  Definition 12.4, G.8
  Syntaxüberprüfung 6.4
Interpreter G.8

K

Klammern, zum Festlegen des Vorrangs verwendet 4.10
Klicken G.8
Kommentare, die in QKARTEN erscheinen 5.4
Kompilieren G.8
Konstanten
  Definition G.8
  lokale G.9
  symbolische G.14
Kontextabhängige Hilfe 3.12
Kontrollfelder
  Abbildung 2.12, 10.15
  Beschreibung 2.12, 10.15
  Definition G.8
Konventionen, Dokument xix
Koordinaten
  absolute G.1
  physikalische G.12
Kopieren
  Beispiel aus Hilfe 8.4
  Text 12.11, 13.4
Kopieren, Befehl 6.10, 13.4
Koprozessor 16.11, G.8
Kurze Adresse G.9

Kurzmenüs
auswählen der Befehle 2.7-8
Beschreibung 2.1
mit welchen Fenstern verfügbar 3.1
Vergleich mit vollständigem Menü 20.7

L

/L-Option (QB) 10.3
Laden
Dateien 11.15
Module 11.12, 11.15, 16.12
Lange Ganzzahlen G.9
Laufwerke G.1
Laufzeit
Definition G.9
Fehler
Hinweise betreffend 17.4
Positionen anzeigen 16.8
simulieren 10.25
überprüfen 16.10
Laufzeitbibliotheken
ausführbare Dateien 16.6
Definition G.9
Laufzeitmodul G.9
in Programmen verwendetes 16.9
Leeren des Arbeitsfensters 3.6
Leertaste, verwenden in Dialogfelder 2.12
Lehrmaterial verwenden 1.3
Lernen.com xviii
Liesmich.doc xviii
Linker, BRUN45.exe, Vorteil von 16.9
Logische Bezeichungen 4.18
Logische Geräte G.9
Logische Operatoren 4.20
Lokale
Konstanten G.9
Symbole G.10
Variablen G.9
Löschen
Arbeitsfenster 3.6
Bildschirme 10.7
Haltepunkte 9.2
Module 11.18
Text, Befehlszusammenfassung 12.11
Variablen aus dem Debug-Fenster 9.2
Löschen, Befehl 13.6
LPT1 G.10

M

.mak-Datei 16.16
Marke, Befehl 15.8
Marken G.10
Markierter Text, Befehl 15.4
Mathematische Funktionen 4.10
Mathematischer Koprozessor G.10
Maus
Cursor G.10
Definition G.10
Kompatibilität mit QuickBASIC 1.4
QuickBASIC, Verwendung mit 10.27
rechte Maustaste setzen 20.5
rollen von Text mit 10.28
verwenden mit Hyperverbindungen 10.31
Mauscursor 10.5
/MBF-Option (QB) 10.3
MCGA *Siehe* Multicolor Graphics Array
MDA *Siehe* Monochrome Display Adapter
Mehrdimensionale Datenfelder G.10
Mehrmodulige Programme 19.1
Menüs
ausführen 16.7
auswählen 10.7
Datei
Abbildung 10.7
Befehle (Liste) 11.1-2
Hilfe 2.5, 21.1
installieren 1.1-2
Kurzmenüs
Beschreibung 2.1
Vergleich mit vollständigem Menü 20.7
öffnen 2.4
Optionen 20.1
Menüzeile 2.2, 10.7, G.10
Metabefehle
$INCLUDE 14.10, 16.16
Definition G.10
Microsoft Binärformat G.10
Microsoft Overlay Linker (LINK) 16.6
Modi
Einfüge- 12.1, G.6
Überschreibe- 12.1
Modul-Ebenen-Code
Beschreibung 5.7
Definition G.11
Modul-Ebenen-Code 5.7, G.11
Modulare Programmierung G.11
Module
ausdrucken 11.19
bestehende 11.15
Definition 11.14, G.11
Module (*Fortsetzung*)
Hauptmodul
ändern 16.15
Definition G.7
laden und entfernen 16.12
Laufzeit G.9
löschen 11.18
neue 11.12
Module entfernen 16.12
Monitor, monochrom
einsetzen mit Farbgraphikadapter 10.2
QuickBASIC starten 2.1
Monochrome Display Adapter (MDA) G.10
Multicolor Graphics Array (MCGA) G.10

N

Nächste Anweisung festlegen, Befehl 17.7
Nächste Anweisung, Befehl 14.7
Nächste SUB, Befehl 14.5
Namen
Basis G.3
Gerät G.7
Pfad G.11
Neue FUNCTION, Befehl 13.11
Neue SUB, Befehl 13.7
Neues Programm, Befehl 3.6, 11.2
Neusetzen von Variablen 16.3
Neustart, Befehl 9.2
Nicht anzeigbare Meldung 17.5
NOT-Operator 4.21
Null-Zeichen G.11
Numerischer Koprozessor *Siehe* Koprozessor
Num-Feststell-Anzeiger
Abbildung 10.6
Beschreibung 2.3
nur zu lesende Datei G.11

O

Objektdateien
Definition G.11
kompilieren mit BC 16.6
Öffnen
Programme 3.3
Offset G.11
On-Line-Hilfe *Siehe* Hilfe

Operand G.11
Operatoren
  arithmetische 4.8
  Definition G.11
  logische 4.20
Optionsflächen
  Abbildung 2.12
  Beschreibung 2.11, 10.15
Optionsflächen 2.12-13
Overlay G.11

P

paket.lst xix
Palette G.11
Parallelanschlüsse G.11
Parameter
  Definition G.11
  formale G.6
Pfadangabezeile 10.14
Pfadnamen G.11
Pfeiltasten *Siehe* Richtungstasten
Physikalische Koordinaten G.12
Potenzierung G.12
PRINT-Anweisung
  kombinieren 4.14
  numerische Ausdrücke 3.7
  Tab-Abstände setzen 4.14
  verwenden 4.2, 4.12
Programm laden, Befehl 11.4
Programm-Befehlsfläche 7.3
Programmablauf, erfassen 18.11
Programmanweisungen wiederholen 4.23
Programmanweisungen 4.2
Programmausführung
  Debug-Befehle 9.2
  Fehler in 6.6
  neue 3.7
Programmausführung, rechte Maustaste 20.5
Programme
  Ablauf verfolgen 18.11
  Ausführen bis Cursor 7.7, 9.3
  ausführen 3.7, 9.2, 10.23
  Ausführung
    fortsetzen 9.2
    unterbrechen 7.9
  Ausgabe 3.7
Programme (*Fortsetzung*)
  auswählen 3.5
  BASICA, laufen mit QuickBASIC 16.2
  beenden 11.21
  Beispiel auf Festplatte xix
  bestehende Programme starten 11.4
  Blöcke einrücken 8.4
  Definition 4.2, G.12
  erforderliche Laufzeitmodule 16.9
  fortsetzen nach Unterbrechung 9.6
  fortsetzen 18.17
  installieren 1.1
  Instruktionen 4.2
  mehrere Module 19.1
  neues Programm starten 11.2
  Neustart bei 1. ausführbaren Answeisung 9.3
  Neustart 9.2, 18.17
  Programmausführung anhalten 18.13
  Programmausführung unterbrechen 7.4
  Schritte G.12
  selbständige 16.9
  starten 18.17
  strukturierte 5.7-9
  testen 10.22
  Variablenwerte verändern in 16.3
Programme anhalten 16.2
Programmierumgebung, Beschreibung xv
Programmierung, modulare G.11
Programmsymbole *Siehe* Symbole
Prozeduren
  alle Anweisungen ausführen 9.3
  Ändern des Standard-Datentyps 13.10
  ansehen 14.3, 14.5
  Arbeitsfenster, Abbildung 9.5
  Aufruf aus dem Direkt-Fenster 10.23
  benennen 13.10
  Beschreibung 5.7
  Definition G.12
  deklarieren 13.10
  erstellen FUNCTION 13.11
  erstellen SUB 13.7
  erstellen 9.3
Prozeduren (*Fortsetzung*)
  Fehlerbeseitigung 9.6
  Folge 19.1
  Standard-Datentypen 13.9
  SUB G.14
  verschachtelte 19.1
Prozedurenaufruf G.12
Prozedurenfolge 19.1
Prozedurschritt, verfolgen 17.8
Prozedurvereinbarungen, automatische 6.8, 9.12
Puffer G.4
Puffergröße für empfangene Daten, Option (QB) 10.3
Punkte zur Textmarkierung
  Definition G.12
  verwenden 12.3
Punkte, Markierungsbefehl 2.8

Q

QB, Befehl 10.2-3
QB-Express xix, 1.3
QB-Ratgeber auf Diskette 10.33
QB-Ratgeber 10.30
  Siehe auch Hilfe
qb.ini, Datei 2.15, 20.1
QKARTEN-Programm 5.3-4
Quelldatei G.12
Quelldateien G.12
Quick-Bibliothek laden, Option (QB) 10.3
Quick-Bibliotheken *Siehe auch* Bibliotheken
  Dateibenennung 16.13
  Definition G.12
  frühere Quick-Bibliotheken aktualisieren 16.12
QuickBASIC
  Fehlerbeseitigung 17.3
  installieren 1.1-2
  Maus verwenden 1.4
  Optionen 20.1
  Schlüsselwörter 21.1
QuickBASIC installieren 1.1-2
QuickBASIC, Schnell laden und speichern, Option 11.11

QuickBASIC-Umgebung, zurückkehren zu 7.9
RAM-Platz, speichern mit ausführbaren Dateien 16.9

R

Ratgeber *Siehe* QB-Ratgeber,
rechte Maustaste 20.5
Rechte Maustaste, Befehl 20.5
Referenz, Übergabe als G.15
Referenzzeile
  Abbildung 2.3, 10.6
  Beschreibung 2.3, 10.7
  Definition G.13
Register G.13
Rekursion G.13
Reservierte Wörter *Siehe* Schlüsselwörter
RETURN ohne GOSUB, Fehlerursache 16.10
RETURN-Anweisung 16.10
Richtungstasten
  Definition G.13
Richtungstasten
  verwenden mit Menüs 2.4
Richtungstasten 10.32
Rollen
  Definition G.13
  Tastaturbefehle 10.18
  Tasten, Zusammenfassung 12.11
  Text in einem Fenster 3.6
Rollen von Text,
  Maus 10.28
Rückgängig machen, Befehl 13.2
Rückverfolgen ein, Befehl 17.4
/RUN-Option (QB) 10.2

S

Schleifen
  DO 4.25
  FOR...NEXT 4.23
Schleifenvariablen 4.23
Schleifenzähler 4.23
Schließen, Hilfe 10.29
Schließen, Menüs, ESC-Taste 10.9
Schlüsselworte
  Definition 4.4, G.13
  Großschreibung von 4.3
  Hilfe 7.2, 21.1, 21.3
Schlüsselwörter
  Großschreibung von 12.8
Schnittstelle
  parallel G.13
  seriell G.13
Schnittstelle G.13
Schreibfehler 12.5
Schwarz-weiß Monitor *Siehe* Monitor, monochrom
Segment G.13
Seitenverhältnis G.13
Selbständige Bibliotheken *Siehe* Bibliotheken
Selbständige Dateien G.13
Selbständige EXE-Datei, Option 16.8
Selbständige Programme, Vorteile 16.9
SELECT CASE-Anweisung
  Beschreibung 8.8
  Hilfsanzeige, Abbildung 8.5
Sequentielle Dateien G.13
Serielle Anschlüsse G.13
Setup-Menü 1.1-2
Setup-Programm 1.1
SHARED 17.6
Software, Probleme berichten xxii
Sonderzeichen, Methoden zur Eingabe 12.8
Spaltenzähler, Abbildung 10.6
Speichern
  .mak-Dateien 16.16
  Dateien 11.9-10
  Prozeduren 13.10
Speichern Programme 5.12
Speichern unter, Befehl
  Beschreibung 11.11
  Dialogfeld, Abbildung 3.15, 11.11
  QuickBASIC-Option 11.11
  schnell laden und speichern, Option 11.11
  Text-Option 11.11
Speichern, Befehl 11.9
Sprachhilfe 10.30
Standard G.13
Standard-Datentypen
  ändern in Prozeduren 13.10
  für Prozeduren 13.9
Standard-Eingabe und -Ausgabe G.14
Standard-Suchpfade 20.3
Stapel
  Aufrufe, Menü 19.4
  Definition G.14
Start, Befehl 3.7, 9.2
Starten *Siehe auch* Laden
  Programme
    bei gegebener Anweisung 18.17
    bestehende 3.3, 11.4
    neue 11.2
  QuickBASIC 2.1, 10.2
Statische Datenfelder G.14
STEP-Anweisung 4.24
Steuerfolge-Zeichen 12.9
Stoppbedingungen 17.4-5, G.14
Stoppbits G.14
STRG+UNTBR 16.10
STRG-Taste, Sonderzeichen eingeben 12.8
Strukturierte Programmierung 5.7-9, 12.2
SUB-Prozeduren
  ansehen 14.1
  benennen 13.10
  Definition G.14
  erstellen 13.7
  Standard-Datentyp 13.9
SUBs, Befehl
  Beschreibung 14.1
  Dialogfeld, Abbildung 14.3
  Prozeduren verschieben 5.9, 9.5, 13.7
SUBs, Befehl 5.10
Suche, Befehl
  Beschreibung 15.2
  Dialogfeld, Abbildung 15.2
  Einschränkungen 15.3
  Optionen 15.3
Suchen
  Marken 15.7
  Prozeduren 14.3
  Text 6.12, 15.1, 15.5

Suchen, Menü
  Ändern, Befehl 15.7
  Befehle (Liste) 15.1
  Marke, Befehl 15.8
  Markierter Text, Befehl 15.4
  Suche, Befehl 15.2
  weitersuchen, Befehl 15.5
Suchpfade festlegen, Befehl 10.32-33, 20.3
Suchpfade, Befehl 20.3
Symbole
  Definition G.14
  globale G.7
  Hilfe 108
  lokale G.10
  verwenden 7.6
Symbolhilfe 9.10
Symbolische Konstanten G.14
Syntax
  Fehler, überprüfen 12.4-5
  Hilfe 8.1, 21.3
  überprüfen 6.4, 12.4, 20.6
Syntaxüberprüfung, Befehl
  ausschalten 12.5
  Beschreibung 12.4
  Einführung 20.6

T

Tab 12.3
Tab-Abstand, setzen 2.13, 12.2, 20.2
TAB-taste, verwenden in Dialogfelder 2.11, 2.13
Tastatur
  Befehle wählen über 10.8
  bewegen im Hilfsfenster 10.32
Tasten
  Direktkombinationen
    Tabelle 10.10-12
    Menü Bearbeiten 2.14
  F10 9.3
  F7 7.7, 9.3
  F8 9.3, 9.8
  F9 9.8
  Hilfe 10.32
  Richtungstasten G.13
  STRG+UNTBR 7.9
  Tastenkombinationen *Siehe auch* direkte Tastenkombinationen
  Beschreibung 2.8
  Bildlauf 3.6
  Cursorbewegung 2.14
  Tastenkurzkombinationen A.1-3
Teilen, Befehl 10.16, 14.6
Testen, Programme 10.22
Text
  ändern 15.6
  auswählen 6.9, 12.2
  Definition G.14
  Editoren G.14
  einfügen 6.10, 8.12, 12.1, 13.5
  eingeben 12.1
  einrücken 6.11, 12.2
  entfernen 13.3
  ersetzen 6.12-14, 13.3
  kopieren 13.4
  löschen 6.12, 13.6
  Option, Speichern unter, Befehl 11.11
  suchen nach 6.12, 15.1, 15.5
  suchen 15.2
  überschreiben 6.7, 12.1
  verschieben 6.11-12, 13.3
Text ersetzen 6.12-14
  Siehe auch Text einfügen
Text-Zwischenspeicher 6.9, 13.2, G.17
Textdateien 11.14, G.14
Textfelder
  Abbildung 10.14
  Beschreibung 2.10, 10.14
  Definition G.14
  Fehlerbeseitigung 10.14
Textfelder 2.10, 10.13
Themen, Befehl 2.7, 21.2
Titelleiste
  Abbildung 2.3
  Beschreibung 10.6
  Definition G.14
Typen G.15
Typographische Konventionen *Siehe* Dokumentkonventionen

U

Überblick, Befehl 2.7, 21.5
Übergabe als Referenz G.15
Übergabe als Wert G.15
Überlauf G.15
Überprüfen
  Syntax 6.4, 12.4, 20.6
  Typen G.15
Überschreiben von Text 6.7, 12.1
Umbenennen von Programmen 5.13
Umgebungs/Compiler-Unterschiede 16.11
Umlenkung G.15
Umschalt/Num-Feststell-Anzeiger
  Abbildung 10.6
  Beschreibung 2.3
Umschalten aktives Fenster 10.17
Umschalter 2.8, G.15
Umwandlung, arithmetische G.2
Ungelöster Verweis G.15
Unterbrechen der Programmausführung 7.4, 7.9
Unterbrechen der Programme 7.9
Unterbrechungen, Prüfung während der Laufzeit 16.10
Unterlauf G.15
Unterprogramme
  Definition G.15
  erstellen 13.7-9
Unterroutinen G.15
UNTIL-Schlüsselwort 4.26

V

Variable anzeigen, Befehl 3.11, 9.2, 17.5
Variablen
  Anzeige 17.5
  anzeigen 4.12-13
  automatische G.3
  Benennungskonventionen 4.3-5
  Definition G.16
  einfache G.6
  ganzzahlige 4.5
  Gleitkomma 4.5

Variablen (*Fortsetzung*)
  globale G.7
  Hilfe 21.3
  Initialisierung 4.4
  lokale G.9
  löschen aus dem Debug-Fenster 9.2
  neusetzen 16.3
  Standard-Typ 4.6
  Typen 4.5
  unzulässige Zuweisung 4.8
  Werte
    aktuelle 9.2
    ändern 3.11, 4.8
    bestimmen 7.6
  Zeichenketten 4.6
Vereinbarungen
  automatische Prozeduren 9.12
  Definition G.16
Verfolgen
  Einführung 17.4
  Programmablauf 18.11
  Prozedurschritte 18.11
  rückwärts 18.12, 18.15
  vorwärts 18.13, 18.15
Verfolgen durch Hilfsanzeigen 10.30
Verfolgen ein, Befehl 17.4
Verfolgen, Ereignisse G.6
Vergleichsanweisungen 4.18
Vergleichsoperatoren 4.18
Verkacheln G.16
Verlassen von QuickBASIC 3.15, 11.21
Verlassen von QuickBASIC *Siehe* Beenden von QuickBASIC
Verschachtelte Include-Dateien 14.10
Verschachtelte Prozeduren 19.1
Verwendung der Hilfe 10.29
Verzeichnisfelder
  Beschreibung 2.10, 10.14
  Bildlaufleiste, Abbildung 10.14
  Definition G.1
Verzeichnisfelder 2.10, 10.13
Verzeichnisse
  aktuelle G.1
  Definition G.16
  Listing 11.6
  Suchpfade ändern 20.4
  wechseln aus QuickBASIC heraus 11.7
  Wurzelverzeichnis G.16
Verzeichnisse auflisten 11.6
Verzweigen 4.22
Video Graphics Array (VGA) G.16
Vollständiges Menü 20.7
Vorhergehenden Suchvorgang
  wiederholen *Siehe* Weitersuchen, Befehl
Vorrang
  Definition G.16
  Einführung 4.9

W

Wahr, Darstellung 4.20
Weiter, Befehl 9.2, 9.6
Weitersuchen, Befehl 15.5
Werte verändern in laufenden Programmen 10.24
Werte, übergeben an G.15
WHILE-Schlüsselwort 4.26
WordStar-Stil, Befehlsanzeiger, Abbildung 10.7

Z

Zeichenketten
  Definition 4.3, G.16
  kombinieren 4.7
  Übergabe an QuickBASIC 16.5
  Zeichen G.16
Zeichenketten G.16
Zeichenkettenvariablen 4.6
Zeiger G.16
Zeilen, setzen von Attributen 2.13
Zeilen- und Spaltenzähler 2.3
Zeilennummer prüfen 16.10
Zeilenzähler, Abbildung 10.6
Ziehen G.16
Zusammenführen, Befehl 11.7
Zuweisung G.16
Zuweisungsanweisungen 4.7
Zweierkomplement G.16
Zweimal klicken G.17

Microsoft®

1288 Artikelnr. 06026

Microsoft®

1288 Artikelnr. 06026

Microsoft QuickBASIC

Lernen und Anwenden von Microsoft QuickBASIC

Microsoft® QuickBASIC

Lernen und Anwenden von Microsoft QuickBASIC

Microsoft®